А. М. Елизаров
Е. К. Липачев
М. А. Малахальцев

ОСНОВЫ MathML. Представление математических текстов в Internet

Практическое руководство

ОСНОВЫ MathML. Представление математических текстов в Internet

Практическое руководство

А. М. Елизаров, Е. К. Липачев, М. А. Малахальцев

Казань
Издательство КМО
2008

Казанское математическое общество
Российская Федерация,
Татарстан, 420008, Казань,
Ул. Профессора Нужина, 1/37
НИИ математики и механики
им. Н. Г. Чеботарева
Казанского государственного
университета

Kazan Mathematical Society
Chebotarev Institute of
Mathematics and Mechanics
Kazan State University
1/37 Professor Nujin Str.
Kazan, Tatarstan,
420008,
Russian Federation

e-mail: kmf@ksu.ru

Российский фонд фундаментальных исследований

Издание осуществлено при финансовой поддержке
РФФИ (проект № 06-07-89132)

УДК 004.912: 004.738.5
ББК 81.1

Печатается по постановлению Редакционно-издательского совета Казанского математического общества

Основы MathML. Представление математических текстов в Internet. Практическое руководство/ *А. М. Елизаров, Е. К. Липачев, М. А. Малахальцев*. 2-е изд., переботанное и дополненное. – Казань: Издательство Казанского математического общества, 2008. – 100 с.

Руководство содержит рекомендации по подготовке математических текстов для публикаций в электронной форме с использованием языка MathML. Описаны технология MathML и программные средства, позволяющие конвертировать в MathML документы, подготовленные с помощью имеющихся стандартных технологий (LaTeX, Mathematica, Maple, Word).

Для научных работников, преподавателей, аспирантов и студентов, специализирующихся в области естественных наук.

УДК 004.912: 004.738.5
ББК 81.1

ОГЛАВЛЕНИЕ

Введение

В наши дни большая часть информации поступает, хранится и перерабатывается в электронном виде. Заметные изменения произошли и в сфере научного обмена, в результате современного исследователя невозможно представить без компьютера, который из средства набора текстов давно стал инструментом получения знаний. Поскольку электронная форма представления информации стала основной, для математиков проблема представления и обработки математических текстов в электронной форме приобрела особую актуальность. В существенной части эта проблема касается представления математических формул. Представление математических документов в pdf-формате или html-файлах, когда каждая формула является ссылкой на графический ресурс, делает структурную обработку такой информации крайне затруднительной. Поэтому наиболее распространенное на данный момент решение – представление формул в виде графических файлов – неудовлетворительно с точки зрения структурной обработки математических текстов.

Международная организация World-Wide Web Consortium (сокращенно W3C, см. сайт www.w3.org), разрабатывающая технологии глобальной сети, предложила новую концепцию развития интернета – Semantic Web (семантический веб), направленную на изменение основных принципов функционирования всемирной сети. Главная ее цель – обеспечение машинного управления информационным пространством. В своей основополагающей работе [1] («Дорожная карта семантического веба») Тим Бернерс-Ли утверждает: «Веб разрабатывался как информационное пространство, полезное не только для коммуникации человека с человеком, но и как пространство, в котором эффективное содействие могут оказывать также и машины; … подход семантического веба базируется на разработке языков для выражения информации в форме, пригодной для машинной обработки».

Как указано в программных документах консорциума W3C [2], семантический веб – это «...расширение традиционного веба в направлении существенно лучшего определения смысла информации, позволяющего компьютерам и людям эффективнее выполнять совместную работу. Мы хотим, чтобы данные на вебе были определены и связаны ссылками так, чтобы их можно было легче находить, интегрировать, автоматизировать и повторно использовать в различных приложениях, ... чтобы данные были разделяемыми и могли обрабатываться как автоматизированными средствами, так и людьми». Конечная амбициозная цель этого проекта состоит в создании такой среды, где программные агенты могут динамически обнаруживать и опрашивать ресурсы, а затем взаимодействовать с

ними. Такие агенты должны уметь справляться с возникающими виртуальными проблемами интеллектуализированной среды, обнаруживать новые факты и выполнять изощренные задания, получаемые от людей (см., например, [3]).

Для математического сообщества наибольший интерес представляет MathML (Mathematical Markup Language) – технология, предназначенная для представления математических формул. Разработка этой технологии ведется консорциумом W3C с 1999 года. Поскольку технологии семантического веба предоставляют новый способ организации веб-информации, который дает возможность на более высоком уровне решать задачи программной обработки документов (в частности, задачу поиска), MathML изменяет принципы организации и управления электронными публикациями по математике. В настоящее время язык MathML фактически стал стандартом представления математической информации в электронной форме в силу следующих причин:

- технология обработки данных на основе языка MathML реализует одну из основных тенденций современной информатики – разделение разметки и данных, поэтому она дает широкие возможности многоуровневого структурирования данных и расширенного поиска;
- создано программное обеспечение, использующее технологию MathML; в частности, созданы программные средства, позволяющие конвертировать в MathML документы, подготовленные с помощью имеющихся стандартных технологий (таких, например, как LaTeX, Mathematica, Maple, Word). MathML поддерживается основными браузерами: Internet Explorer (при установке соответствующего плагина), Mozilla, Firefox;
- технология MathML поддерживается системами Maple® и MathCAD 2001, а компания Wolfram Research предложила собственную концепцию использования технологии MathML, которая реализована в пакете Mathematica®, в частности, в этом пакете предусмотрено сохранение документов в формате MathML.

Настоящее издание ориентировано на математиков, заинтересованных в распространении своих результатов в электронной форме, что особенно значимо в связи с тенденцией перехода на онлайновое представление научных результатов. Например, международная инициатива Open Access to Research (http://www.eprints.org/openaccess/) предполагает, в частности, создание онлайновых открытых архивов каждой исследовательской организации (см. [4]). Мы предполагаем, что читатель знаком с основами HTML, желательно также, чтобы он имел представление о системе TeX.

Руководство подготовлено и опубликовано в рамках проекта 06-07-89132

Российского фонда фундаментальных исследований.

1. Семантический веб

Переизбыток и, одновременно, недостаток информации – типичная ситуация, с которой сталкивается современный пользователь интернета. Повсеместное распространение компьютеров во все сферах деятельности и развитие интернета делают доступным все больший объем информации, поэтому отлаженные приемы обработки информации становятся менее эффективными. Поиск нужной информации стал серьезной проблемой: поисковые машины, индексирующие html-страницы, находят много ответов на запрос и охватывают большую часть всего веба, однако количество неудовлетворительных возвращаемых ответов велико, поскольку не существует понятия «правильности» ответов на запросы. Дело в том, что поиск основан на сравнении строки запроса со строками документов, но при этом никак не учитывается смысл информации, ради которой и был организован поиск. Такая ситуация возникла сравнительно недавно, когда развитие интернета привело к тому, что объемы информации, получаемой при поиске, стали намного превышать возможности человеческого восприятия. Отметим, что совокупность технологий первоначальныго интернета, получившая, впоследствии, название Web 1.0, была ориентирована только на формальное содержание документов (контент). В настоящее время используется набор технологий, который принято обозначать Web 2.0 и который также оперирует в основном с контентом [5, 6].

Ведущие производители программного обеспечения, в числе которых Oracle, IBM, Adobe, Sun, Microsoft, Mozilla Inc., в качестве основного направления развития интернета на ближайшие годы разрабатывают новую систему, обозначенную как Web 3.0 и основанную на семантической обработке информации. Разработка этого комплекса технологий координируется консорциумом W3C. Особенность этой системы состоит в том, что программные модули (а не пользователи!), опираясь на метаданные и метабазы, осуществляют поиск информации по содержимому, включая поиск по видео- и цифровым изображениям. Основная задача Web 3.0 заключается в решении самой сложной проблемы развития интернета – поиска значимой информации, отделения её от информационного мусора. Семантический веб позволит рассматривать интернет в целом как глобальную базу данных (БД). Точно так же, как разработчик может запрашивать сведения из обычной БД и создавать приложения, оперирующие этой информацией, любой человек получит возможность собирать данные во всей интернет-сети и в соответствии со своими нуждами строить приложения, обра-

батывающие взаимосвязанные, но разрозненные сведения из различных источников [7]. Это соответствует программному заявлению Т. Бернерса-Ли: «... основной ролью технологий семантического веба является интеграция данных, содержащихся в различных приложениях».

Название Semantic Web появилось в 2001 году в статье [8], аналогичное название получил проект консорциума W3C. В русскоязычной компьютерной литературе в последнее время используют термины «семантический веб», «семантическая сеть» и «семантическая паутина». Слово «семантика» дало название проекту и определило общее направление развития.

Общее определение понятия «семантика» – это изучение значений. Слово «семантика» происходит от греческого semantikos, т. е. «важное значение». Компьютер должен понимать семантику документа в том смысле, что он не просто интерпретирует набор символов, содержащихся в документе, а выделяет смысл документа.

Имеется несколько определений понятия Semantic Web, наиболее подходящим из которых, на наш взгляд, является определение, приведенное в электронной энциклопедии Википедия (см. [5]): «*Семанти́ческая паути́на* – часть глобальной концепции развития сети Интернет, целью которой является реализация возможности машинной обработки информации, доступной во Всемирной паутине. Основной акцент концепции делается на работе с метаданными, однозначно характеризующими свойства и содержание ресурсов Всемирной паутины, вместо используемого в настоящее время текстового анализа документов. В семантической паутине предполагается повсеместное использование, во-первых, универсальных идентификаторов ресурсов (URI), а во-вторых – онтологий и языков описания метаданных». Предполагается, что семантика способна однозначно охарактеризовать найденный контент по ряду характерных признаков. Архитектура семантической сети предполагает наличие у любой информации, находящейся в сети, связанного с этой информацией точного смысла, который нельзя перепутать даже в случае совпадения фраз или слов, встреченных в разных контекстах. Фактически это означает, что любая информация связана с некоторым неотделимым от нее контекстом [9]. Семантический веб использует несколько основных технологий для выявления смысла данных.

Для трактовки данных он использует универсальный идентификатор ресурсов (URI). При этом URI рассматривается в более широком смысле – не только как ссылки на электронные адреса и веб-страницы, как в традиционной схеме, но и для обозначения любых объектов и ресурсов (люди, города, предметы и др.). Сейчас большая часть информации в сети совершенно не приспособ-

лена для компьютерной обработки, поэтому не удалось создать программы, которые были бы способны разобраться в смысловой составляющей текста и, например, сгруппировать несколько текстов в одну общую категорию. В семантической паутине предлагается использовать форматы описания, доступные для машинной обработки и позволяющие решить эту задачу.

Для определения собственной структуры документов в семантической сети используют язык XML (eXtensible Markup Language), а для формализации метаданных, а также сведений о контексте – RDF (Resource Definition Framework). Для записи конструкций RDF можно использовать RDF/XML, являющийся подмножеством языка XML, а для описания структуры документов – RDF Schema. Для построения семантически связанной сети недостаточно только технологий XML и RDF, поэтому консорциумом W3C и был создан язык онтологии OWL (Ontology Web Language). Возможность OWL создавать онтологии играет ключевую роль в категоризации и классификации групп взаимосвязанных (related) данных [7].

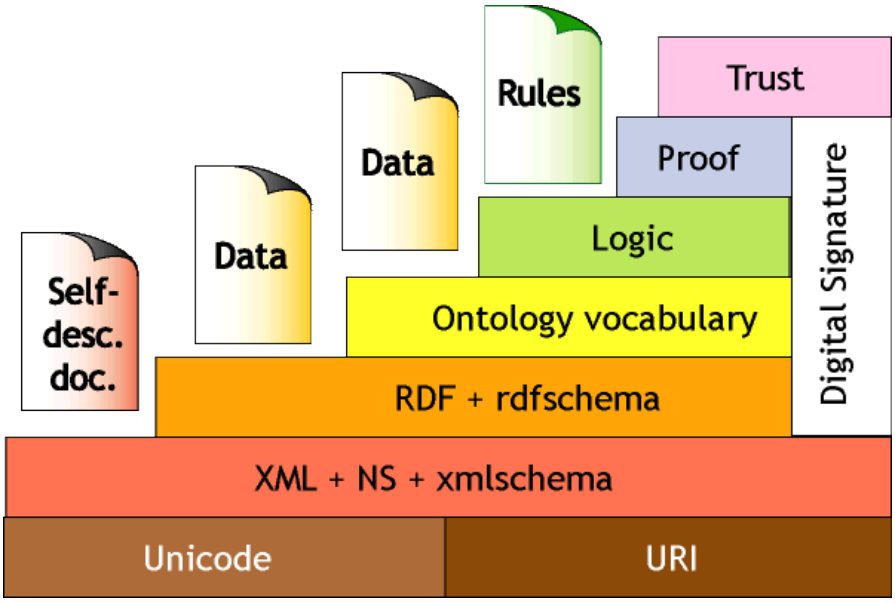


Рис. 1. «Пирог Тима Бернерса-Ли»

Семантический веб создается как надстройка над уже существующими системами сетей, при этом поиск и обработка информации программируются как машиноориентированные. Чтобы это стало возможным, производится дублирование содержания контента в метабазы. Информация, предназначенная для людей, готовится в виде текста, образов и звуков, а для машин – в виде специальных кодов. Семантический веб объединяет эти виды информации в единую структуру, в которой каждому элементу «человеческой» информации будет соответствовать машинный код – специальный смысловой тэг. Метаданные должны в обязательном порядке включать сведения о том, как, где и кем была собра-

на данная информация и как она структурирована.

При описании многослойной архитектуры семантической сети обычно используют диаграмму, впервые предложенную Т. Бернерсом-Ли в презентации [10] и получившую название layer-cake, а в русскоязычной литературе – «пирог Тима Бернерса-Ли» (см. рис. 1). Дадим краткое описание этих слоев.

Первый слой «Unicode и URI». Unicode – это стандартная кодировка для представления символов. URI (Uniform Resource Identifier) – универсальный идентификатор ресурсов – это просто идентификатор ресурсов, с которыми мы постоянно сталкиваемся при работе в интернете (например, http://www.ksu.ru).

Слой XML и связанных с ним стандартов. XML предоставляет синтаксис для определения структуры документа, подлежащего машинной обработке. Синтаксис XML не несёт семантической нагрузки. XML Schema определяет ограничения на структуру XML-документа. Стандартный синтаксический анализатор языка XML в состоянии проверить произвольный XML-документ на соответствие его структуры так называемой схеме документа, описанной в XML Schema. «Схема» – это просто документ или фрагмент кода, управляющий множеством терминов в другом документе или фрагменте кода, как главный контрольный список или грамматика определений.

RDF представляет собой простой способ описания экземплярных данных в формате субъект – отношение – объект, в котором в качестве любого элемента этой тройки используются только идентификаторы ресурсов. Существует стандартизованное отображение этих троек на XML-документы предопределённой структуры (т. е. консорциумом W3C определена схема XML-документов, содержащих RDF-описания), а также на другие форматы представления (например, в нотацию NS). Схема RDF была разработана как простая модель типизации данных для RDF. RDF Schema описывает набор атрибутов (здесь их точнее назвать отношениями) для определения новых типов RDF-данных.

Слой онтологий (Ontologe vocabulary). Онтологии предназначены для описания более сложных конструкций, включающих в себя типы ресурсов и их свойства. Язык онтологий веба OWL расширяет возможности по описанию новых типов (в частности, добавлением перечислений), а также позволяет описывать новые типы данных RDF Schema в терминах уже существующих (например, определять тип, являющийся пересечением или объединением двух существующих).

Слой логики и доказательства (Logic and Proof). Логический слой необходим как средство для формулировки в документах логических выражений. Это позволяет, например, записывать правила вывода документов одного типа из до-

кументов другого типа; проверять соответствие содержимого документа некоторому множеству правил непротиворечивости; преобразовать запросы для замены неизвестных терминов известными. Примером приложения этого уровня является преобразование запросов к одной базе данных в запросы к другой базе данных в случае, когда базы данных на вебе построены независимо и объединены с помощью семантических ссылок.

Слой управления доверием (Trust) – это завершающий слой. Этот компонент находится еще на этапе разработки, одним из его элементов является, например, технология цифровой подписи.

Как известно, реализованная машина логического вывода должна быть тесно связанной с системой верификации подписей, а грамматический разбор документов должен давать в результате не просто деревья утверждений, а деревья утверждений о том, кто и какие утверждения подписывал. При проверке доказательств для правил вывода должна проверяться логическая корректность, но для утверждений о наличии подписи в документе должна проверяться корректность этой подписи. В конечном счете, мы должны получить структуру, в которой можно выражать и вычислять соотношения доверительности и надежности на всем множестве систем, базирующихся на открытом ключе.

Как результаты, так и способы их получения одинаково ценны. Откуда мы получаем данные, кто их создал, когда, почему и как они были получены – так же важно для пользователя и провайдера службы, как и сами данные (см. [11]). Этими вопросами в семантическом вебе занимаются такие приложения, как Доказательства, Доверие и Цифровые подписи. Возможными приложениями семантического веба и технологиями баз данных будут приложения по сопровождению аннотаций комментариями и ссылками на источники, по объяснению, почему выбранные параметры использовались при работе алгоритма и каковы эти параметры.

Научное знание контекстуально и субъективно. Контексты изменяются, а мнения противоречивы. Новая информация может как поддерживать господствующую точку зрения, так и противоречить ей, приводя к ревизии понятий. Анализ этих аналитических высказываний может привести к появлению нового знания, но этот анализ должен быть доступен для всех, иначе им будет невозможно воспользоваться.

Для надежного хранения научные знания дублируются и архивируются. Очень важно уметь представлять себе срез состояния умов в данный момент в данном месте для осознания научной точки зрения, господствующей в это время. Однако по мере развития коллекций данных и аналитических приложений

прослеживать последствия изменений становится все труднее. При модификации баз данных ученые вынуждены заново формулировать свои запросы, новые знания изменяют фактическую область анализа. Ошибки или информация, потерявшая достоверность, продолжают распространяться и с трудом поддаются элиминации. Онтологии и правила также изменяются. Новые верования заставляют изменять онтологии, но прежние выводы и заключения, перестающие быть справедливыми, при этом не исчезают автоматически (и как вообще мы следим за влиянием этих изменений?). Они должны продолжать сосуществовать и быть доступными. Мониторинг событий и сущностей может быть описан с помощью онтологий.

Графический способ представления основных технологий семантического веба, предложенный Т. Бернерсом-Ли, получил развитие – можно найти схемы с другой организацией слоев. В частности, в Википедии [5] приведена схема, представленная на рис. 2.

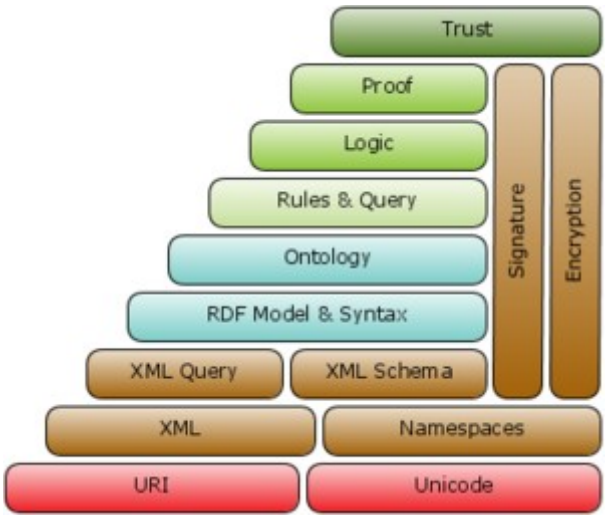


Рис. 2. Стек понятий семантической паутины из Википедии

Т. Бернерс-Ли предложил следующий способ определения, действительно ли в том или ином продукте реализована технология семантического веба: следует лишь обратить внимание на поддержку стандартов. Если продукт не поддерживает такие основополагающие стандарты, как RDF, OWL или SPARQL, то он к семантическому вебу не имеет никакого отношения [7]

Необходимость развития технологий Web 3.0 разъясняется в следующем фрагменте из работы [12]: «Учитывая эти факты, трудно оспаривать XML как механизм обмена в Сети. Однако модель XML порождается главным образом самими документами; более точно – текстовыми документами с иерархической структурой. Например, правильно составленный XML-документ может иметь элемент <US-address>, содержащий элементы <US-street-address-line> и <US-zip-code>. С точки зрения «чистого» XML у нас нет ничего, кроме основанного

на данном документе описания имён и заданного в нём расположения этих элементов: а именно, мы знаем лишь, что элемент <US-address> содержит элемент <US-zip-code>, являющийся строкой. Мы вполне можем составить документ, содержащий два элемента <US-zip-code>. В XML-схеме для элемента <US-address> можно достичь большего, задав определённые ограничения на его структуру, например, что этот элемент может содержать несколько элементов <US-street-address-line>, но лишь один элемент <US-zip-code>. Имея эту схему, валидатор сможет вывести, что подобный элемент <US-street-address>, содержащий два элемента <US-zip-code>, является синтаксически неправильным. Однако фактически такая схема не позволяет уловить бóльшую часть семантики понятия «адрес». Например, если мы встретим несколько отдельных документов, в которых указаны адреса, совпадающие полностью за исключением почтового индекса, то мы не сможем распознать, основываясь лишь на описанной схеме, что, по меньшей мере, один из этих индексов является ошибочным. Аналогично, не существует способа задать автоматическое преобразование между двумя очень похожими схемами кодирования адреса (например, если в одной из схем имеется ровно один элемент <US-street-address>, содержащий в свою очередь несколько элементов <US-street-address-line>). Далее, если мы захотим сделать нечто чуть более сложное, например, интегрировать эти элементы <US-street-address> с базой данных о нашем персонале, нам придётся во всём полагаться на человека-программиста. Нужно будет, чтобы кто-то установил правильное отображение объектов <US-address> в различные поля нашей базы данных, причём данное отображение будет чувствительным даже к малейшим изменениям формата наших XML-данных или полей базы данных – не говоря уж о возможности использовать то же отображение в других не связанных с нами системах.

Ещё более удручающим является то, что не существует очевидного способа связать наши данные в <US-address> с более общими фактами об адресах, например, что каждый адрес задаёт некое местоположение, что на них можно отправлять почту, или же что адреса связаны с конкретными людьми или фирмами. И уж заведомо ни XML-модель, ни XML-схема не в состоянии выразить мысль, что почтовые адреса в некотором роде сходны с номерами факсов, по крайней мере, в том смысле, что отправка письма по адресу и факсимильного сообщения по определённому номеру позволяет передать текст послания некоему получателю.

В Сети же нам крайне важна семантика такого рода. Мы хотим уметь привязывать адреса, закодированные с помощью языка XML в некотом веб-доку-

менте, к конкретным людям, местоположениям, понятиям, письмам, другим документам, базам данных, справочникам, спискам адресатов в электронной записной книжке, календарям, сенсорам, услугам и всевозможным другим ресурсам в Сети. Если всё это сделать правильно, то такие связки позволят автоматически обрабатывать информацию, хранящуюся в различных формах и форматах, с помощью компьютерных программ. Безусловно, язык XML является важной составляющей для решения этих проблем, но маловероятно, что XML-модель данных способна подходящим образом представить все эти взаимосвязи и закодированную в них семантику реального мира».

В заключение отметим, что в настоящее время уже работают сайты, созданные по технологии Web 3.0 [5], например,

http://www.sun.com/servers/wp.html/,

http://www.forum.nokia.com/,

http://pressroom.oracle.com/,

http://www.harpers.org/.

2. Кратко об XML и сопутствующих технологиях

2.1. Правильно построенные XML-документы

Всюду ниже повсеместно используется термин *разметка*. Под *разметкой* текста принято понимать расстановку в тексте специальных знаков (*кодов разметки*), с помощью которых осуществляется форматирование текста перед его отображением или печатью. Набор соглашений о разметке называют *языком разметки*. Один и тот же текст можно разметить, использовав разные языки разметки. Например, следующий текст размечен с помощью HTML:

```
<html>
<body background="#FFFFFF">
  <h1>Что такое разметка</h1>
   <p>
     Текст обычно форматируют, выделяя определенные фрагменты курсивом, подчеркиванием и т. д. Это достигается путём расстановки в тексте специальных значков, так называемых <I>кодов разметки</I>.
</body>
</html>
```

Этот же текст можно разметить, например, с помощью LaTeX. Документ, сформатированный в MS Word, также содержит разметку, но она является скрытой.

Одним из первых языков разметки был язык GML (Generalized Markup Language), разработанный IBM в 1960 г. Позже на его основе был разработан SGML (Standard Generalized Markup Language), принятый в качестве международного стандарта в 1986 г. («ISO 8879:1986 Information processing – Text and office systems – Standard Generalized Markup Language (SGML)»). SGML предоставлял множество вариантов синтаксической разметки для использования различными приложениями, однако его сложность затруднила его широкое распространение. Тем не менее, от SGML произошли используемые в настоящее время такие языки разметки, как HTML, XML, SGML Docbook. Далее речь пойдет об XML, который является подмножеством SGML, разработанным для упрощения процесса машинного разбора документа.

В начале февраля 1998 года консорциум W3C рекомендовал спецификацию Extensible Markup Language 1.0, за которой закреплена аббревиатура XML. Прежде всего, отметим, что эта спецификация является основой для построения грамматики языков разметки, и лишь условно сам XML можно назвать языком разметки (в настоящее время создан ряд языков разметки, являющихся подмножествами XML, в частности, язык MathML, который подробно рассматривается ниже). Язык XML разработан Рабочей группой (заглавная буква используется

при описании стандарта и в других документах), образованной в 1996 году. В [13] указаны цели создания этой спецификации, сформулированные в виде десяти заповедей. По своему замыслу технология XML должна обеспечить отделение информации от разметки, что позволяет производить обработку, поиск и представление информации на более высоком технологическом уровне.

На основе этой технологии обработка, поиск и представление информации переходят на совершенно иной уровень. Как отмечено в [1, 14], первоначальный веб был ориентирован на работу человека, но веб следующего поколения (семантический веб) должен в значительной мере опираться на машинную обработку информации, стандарту XML при этом отводится роль одной из ключевых технологий.

Синтаксис XML построен на основе тегов, но в отличие от HTML, в котором множество тегов фиксировано, в рамках XML пользователь создает собственное множество тегов и задает структурные отношения между ними. При создании документов необходимо учитывать правила, основная задача которых – в отделении данных и формата. Эти правила определены в [13] и состоят из следующих требований:

- документы XML должны начинаться с **объявления XML**, в котором определяется версия XML, например, **<?xml version=”1.0”>** (это объявление является инструкцией приложению, обрабатывающему документ, в частности, браузеру)**;** любому открывающему тегу должен соответствовать закрывающий тег;
- в XML учитывается регистр символов;
- значения атрибутов, используемых в определении тегов, должны заключаться в кавычки;
- в любом XML-документе должен содержаться один корневой элемент для всего документа; элементы не должны перекрываться;
- вся информация, размещенная между открывающим и закрывающим тегами, рассматривается как данные, при этом учитываются все символы форматирования (пробелы, конец строки, табуляция);
- в XML имеются зарезервированные символы, которые используются как элементы синтаксиса; в тексте эти символы нужно заменять последовательностями других символов, называемых *объектами* (*entities*; в русскоязычной литературе часто используют термин *сущности*).

Зарезервированный символ	Объект
<	&lt;
>	&gt;
&	&amp;
“	&quot;
‘	&apos;

Если перечисленные правила выполнены, то говорят, что документ *правильно построен (well-formed)*, и только в этом случае он является *документом XML*. Такой документ будет правильно отображен в браузере, в ином случае будет выдано сообщение об ошибке. Для обработки документов XML используется программный модуль, называемый XML-процессором. Поддержка XML браузером зависит от того, включен ли в него XML-процессор и какие возможности этот процессор имеет. Например, браузеры Mozilla и Firefox поддерживают почти все технологии, основанные на XML.

2.2. Пример XML-документа

На простом примере покажем особенности синтаксиса XML. Опишем электронный журнал, причем включим в описание только название журнала и адрес, а также несколько статей с указанием их названия и авторов.

Пример 2.2.1. Листинг файла Example1.xml. Простейшее описание электронного журнала.

```
<?xml version="1.0" encoding="windows-1251"?>
<!-- XML – пример -->
<journal>
<title>Lobachevsky&apos;Journal</title>
<contacts>
<address>Kazan State University</address>
<url>ljm.ksu.ru</url>
</contacts>
<articles>
<article ID="1">
<title>MathML and TeX</title>
<author>M. Malakhaltsev</author>
</article>
<article ID="2">
<title>MathML and RDF</title>
<author>E. Lipachev</author>
</article>
</articles>
</journal>
```

Первая строка приведенного кода является объявлением разметки

(инструкцией приложению, обрабатывающему документ). Атрибут **version** является обязательным и указывает на версию XML, применяемую для структурирования. Рабочая группа XML намеревается давать последующим версиям спецификации номера, отличные от “1.0”. Получая документ, версию которого он не поддерживает, XML-процессор должен выдать соответствующие сообщения.

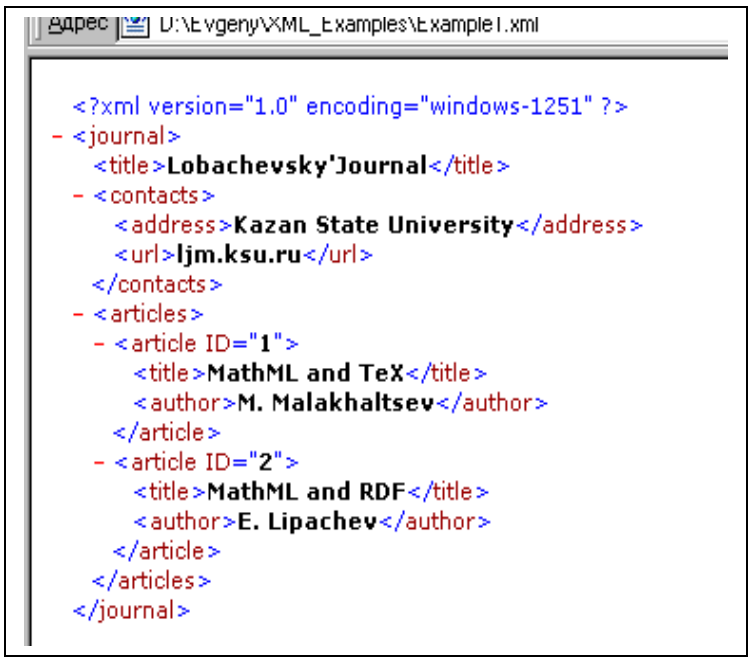
Атрибут **encoding** определяет кодировку документа, и его желательно указывать. Например, если документ содержит символы в одной из кодировок кириллицы (к примеру, CP1251, KOI8-R, CP866), в отсутствие этого атрибута XML-процессор выдаст сообщение об ошибке и прекратит обработку.

Во второй строке показано, как добавить комментарии, – правила те же, что и в HTML.

В последующих строках объявлены элементы. Элемент состоит из открывающего и закрывающего тегов, а также данных между этими тегами. Пустой элемент, например, **
</br>,** может записываться в виде **
. Имена не могут содержать пробелы, кроме того, имеет значение регистр символов. Одни элементы могут быть вложены в другие, как, например, элемент **<author> </author> из приведенного примера вложен в элемент **<article> </article>**, который в свою очередь вложен в элемент **<articles> </articles>**. Один из элементов документа должен содержать все остальные, этот элемент называют *корневым элементом* или *элементом документа*. В нашем примере таким элементом является **<journal> </journal>**. Таким образом, элементы документа должны образовывать древовидную структуру.

У элементов могут быть атрибуты, которые содержат дополнительную информацию об элементе. В приведенном примере элемент **<article> </article>** содержит атрибут **ID**, с помощью которого каждой статье присваивается числовой идентификатор.

Сохраним этот файл с расширением .xml и откроем для просмотра в каком-либо браузере, например, MS Internet Explorer. В результате получим представление XML-документа в виде дерева



Это представление может разочаровать. Для отображения XML-документов в привычном виде используется, например, XSLT-технология (см. раздел 2.6).

2.3. Действительные XML-документы и DTD

С каждым XML-документом можно связать DTD-объявления. Технология DTD (Document Type Definition) разработана как средство описания структуры документа и представляет собой набор инструкций для описания составных частей документа и порядка их следования. DTD определяет множество правил для тегов и атрибутов, допустимых в XML-документе, а также порядок и число вложений тегов, т. е. задается грамматика для определенного типа документов. Для единичного XML-документа создавать DTD не целесообразно, но при разработке некоторого стандарта документа, например, анкеты, желательно написать DTD, чтобы «отсечь» документы, не отвечающие стандарту. О существовании DTD надо помнить и в случае, если XML–документ включается в уже работающую систему без учета грамматики, определённой в DTD этой системы, Ваш документ не будет действительным и не пройдет обработку.

Документ XML имеет три уровня корректности: лексический, синтаксический и семантический. Если документ правильно построен, например, не нарушена вложенность тегов, все теги закрыты и т. д., то он лексически корректен. Синтаксическая корректность означает, что документ соответствует правилам DTD. Таким образом, каждый набор DTD-правил определяет стандарт некоторого класса документов. Например, для документов MathML используется набор правил **mathml2.dtd**.

Инструкции DTD предназначены для XML-процессора (например, встроенного в браузер), анализирующего содержание документа перед его отображением. Если в документе есть ссылка на набор DTD-правил, и он синтаксически им удовлетворяет, то он называется *действительным (valid)* или *корректным*, используется также термин *допустимый* документ.

Инструкции DTD можно записать в отдельном файле или же включить непосредственно в XML–документ. Файл с таблицей DTD является обычным текстовым файлом и может быть создан в любом текстовом редакторе. Для рассматриваемого примера описания электронного журнала содержание DTD-файла может быть таким.

Пример 2.3.1. Файл example1.dtd.

```
<!-- DTD for journal XML -->
<!ELEMENT journal (jrntitle, contacts, articles)>
<!ELEMENT jrntitle (#PCDATA)>
<!ELEMENT contacts (address,url)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT url (#PCDATA)>
<!ELEMENT articles (article)*>
<!ELEMENT article (title, author+)>
<!ATTLIST article
        id ID #REQUIRED>
<!ELEMENT title (#PCDATA)>
```

Для включения внешнего файла DTD в XML-документ используется тег

<!DOCTYPE имя_корня SYSTEM “имя_файла.dtd”>

Имя, указанное за словом DOCTYPE, должно совпадать с именем корневого элемента.

Пример 2.3.2. Файл example2.xml.

```
<?xml version="1.0" encoding="windows-1251"?>
<!-- DTD for journal XML -->
<!DOCTYPE journal SYSTEM "example.dtd">
<journal>
<jrntitle>Lobachevsky&amp;apos;Journal</jrntitle>
<contacts>
<address>Kazan State University</address>
<url>ljm.ksu.ru</url>
</contacts>
<articles>
<article ID="1">
<title>MathML and TeX</title>
<author>M. Malakhaltsev</author>
```

```
</article>
<article ID="2">
<title>MathML and RDF</title>
<author>E. Lipachev</author>
</article>
</articles>
</journal>
```

Определения DTD можно включать непосредственно в XML-файл. Инструкции DTD размещают в начале в контейнере

```
<!DOCTYPE имя_корня [
…
]>
```

Отметим, что блок DTD заканчивается символами **]>**.

Пример 2.3.3. Тот же пример, но с внутренним DTD.

```
<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE journal [
<!ELEMENT journal (jrntitle, contacts, articles)>
<!ELEMENT jrntitle (#PCDATA)>
<!ELEMENT contacts (address,url)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT url (#PCDATA)>
<!ELEMENT articles (article)*>
<!ELEMENT article (title, author+)>
<!ATTLIST article
        id ID #REQUIRED>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
]>
<!-- Конец DTD определений -->
<journal>
<jrntitle>Lobachevsky&amp;apos;Journal</jrntitle>
<contacts>
<address>Kazan State University</address>
<url>ljm.ksu.ru</url>
</contacts>
<articles>
<article ID="1">
<title>MathML and TeX</title>
<author>M. Malakhaltsev</author>
</article>
<article ID="2">
<title>MathML and RDF</title>
```

```
<author>E. Lipachev</author>
</article>
</articles>
</journal>
```

Поясним на основе этого набора DTD-правил наиболее существенные детали DTD. Как уже было отмечено, с помощью правила **<!DOCTYPE** указывается местонахождение DTD. Для описания элементов используется правило

<!ELEMENT имя_элемента содержание>

Для описания содержания используется довольно простая система обозначений. Круглые скобки в содержании означают, что элемент составной. Так, в приведенном примере элементы **journal, contacts, articles** и **article** составные. Запятая в скобках используется в качестве разделителя элементов. Вместо запятой можно использовать символ **“|”** (вертикальная черта), который разделяет варианты содержимого, если их у элемента несколько. Например,

<!ELEMENT phone (#PCDATA|EMPTY)>

определяет, что элемент **phone** может содержать символьные данные (номер телефона) или оставаться пустым (это указано с помощью ключевого слова **EMPTY**). Скобки задают также порядок появления элементов в документе. Например,

<!ELEMENT contacts (address,url)>

указывает, что элемент **contacts** содержит элемент **address,** и этот элемент должен быть первым, а также элемент **url**. Кроме того, можно задать количество повторений элемента, поставив один из знаков **“+”** (плюс), **“*”** (звёздочка) или **“?”** (знак вопроса). Эти знаки можно записывать не только после элементов, но и после скобок – в этом случае действие знака распространяется на множество элементов, заключенных в скобки. Знак “+” означает, что элемент или множество элементов встречаются в документе один или более раз. Знак “*” указывает, что в документе может быть любое число вхождений данного элемента, в частности, элемент может отсутствовать. Знак “?” используется в тех случаях, когда элемент или множество элементов встречаются не более одного раза. Например,

<!ELEMENT article (title, author+)>

указывает, что элемент **article** содержит элемент **title** и он должен быть первым, а также один или более элементов **author**.

Если элемент содержит символьные данные, то при его описании используется содержание **(#PCDATA)**. Ключевое слово **EMPTY** в содержании документа указывает, что это пустой элемент. Как примеры пустых элементов приведём теги **
** и **<hr>** в HTML. Ключевое слово **ANY** в содержании означает, что

элемент может содержать как символьные данные, так и другие элементы.

Если у элемента есть атрибуты, как, например, атрибут **ID** у элемента **article**, то необходимо использовать инструкцию

```
<!ATTLIST    имя_элемента
             имя_атрибута   тип          значение_по_умолчанию
                 ...   ...               ... ... ...
             имя_атрибута   тип          значение_по_умолчанию>
```

В этом объявлении необходимо перечислить все атрибуты, которые могут использоваться с данным элементом. Типы атрибутов можно разбить на три группы: строковые атрибуты, маркированные и перечислимые. Строковые атрибуты описываются ключевым словом **CDATA,** и в документе такие атрибуты могут содержать любые символьные данные. Маркированные атрибуты включают в себя предопределенные атрибуты **ID, IDREF, IDREFS, ENTITY, ENTITIES, NMTOKEN** и **NMTOKENS**. Эти атрибуты предназначены для конкретного использования, например, атрибут **ID** задаёт уникальный идентификатор для элемента в документе. В нашем примере с помощью **ID** каждой статье присваивался числовой идентификатор. Подробности о назначении этих атрибутов можно найти в [15]. Если атрибут содержит список значений, то атрибут называется перечислимым. Например,

```
<!ELEMENT volume (#PCDATA)>
<!ATTLIST   volume
        month (1|2|3|4|5|6|7|8|9|10|11|12) #REQUIRED>
```

Параметр **#REQUIRED** определяет, что данный атрибут является обязательным. Есть ещё два параметра: **#IMPLIED**, указывающий, что атрибут является необязательным, и **#FIXED**, требующий, чтобы только указанное в атрибуте значение использовалось в документе.

2.4. DTD для MathML-документов

Для документов, подготовленных по спецификации MathML, разработано несколько вариантов DTD. Они доступны по адресу http://www.w3.org/Math/DTD/ и имеют следующее назначение:

mathml2.dtd – основная версия (так называемая “stand alone” DTD for MathML);

xhtml-math11-f.dtd – версия, предназначенная для использования в XHTML (“Combined XHTML + MathML DTD”).

При использовании этих DTD можно указать веб-адрес, например,

```
<!DOCTYPE math SYSTEM
“http://www.w3.org/Math/DTD/mathml2/mathml2.dtd”>,
```

или же указать локальное расположение DTD-файла (если он уже «скачан»)

<!DOCTYPE math SYSTEM “mathml2.dtd”>.

2.5. Пространство имен

Пространство имён (namespace) – это некоторое множество имён, терминов, слов, означающих объекты реального мира или концепты. Имена в одном пространстве имён не могут иметь более одного значения. Пространство имён называют также контекстом, так как значение термина может изменяться в зависимости от того, в какое пространство имён он входит. Простым примером является имя человека. В пространстве имён «Семья» его достаточно, чтобы обозначить конкретного человека. В пространстве имён «Граждане Российской Федерации» этого не достаточно – нужно добавить дополнительную информацию – фамилию, адрес и т. п. Пространство имён в программировании – область определения переменных, типов, констант и т. п., которое используется для исключения конфликтов с другими именами вне данного блока.

Спецификация XML Namespaces (пространство имен XML) [16, 17] рассчитана на разрешение проблем, связанных с конфликтами имен. Поскольку множество тегов в XML не фиксировано, как, например, в HTML, пользователь имеет возможность выбирать имена тегов. При подготовке сложных документов, например, использующих несколько DTD, некоторые теги, разные по своему назначению, могут получить одинаковые имена. Поясним эту ситуацию простым примером.

Пример 2.5.1.

```
<?xml version="1.0" encoding="windows-1251"?>
<!-- Конфликт имен title -->
<journal>
<title>Lobachevsky&amp;apos;Journal</title>
<url>ljm.ksu.ru</url>
<articles>
  <article ID="1">
    <title>MathML and TeX</title>
  </article>
  <article ID="2">
    <title>MathML and RDF</title>
  </article>
</articles>
</journal>
```

В этом примере имя **title** используется для тега, содержащего название журнала, и тега с названием статьи. Заметим, что даже при наличии конфликта имен в документе он может без проблем отображаться в браузере, но более

сложная обработка документа может оказаться невозможной.

Пример 2.5.2. Разрешение конфликта имен с помощью Namespaces.

```
<?xml version="1.0" encoding="windows-1251"?>
<journal xmlns:x="http://www.kcn.ru/one"
      xmlns:y="http://www.kcn.ru/two">
<x:title>Lobachevsky&apos;Journal</x:title>
<url>ljm.ksu.ru</url>
<articles>
 <article ID="1">
  <y:title>MathML and TeX</y:title>
 </article>
 <article ID="2">
   <y:title>MathML and RDF</y:title>
 </article>
</articles>
</journal>
```

Как правило, для обозначения пространства имен используются ссылки, образованные по спецификации URL, т. е. с каждым пространством имен связывают некоторый адрес. Механизм образования веб-адресов в силу своей природы должен обеспечить уникальность обозначения пространства имен. Поскольку используется лишь алгоритм образования адреса, нет необходимости в выборе адреса реально существующего веб-сайта, и поэтому адрес можно составить совершенно произвольно. Атрибут **xmlns** используется как ключевое слово XML для обозначения объявления пространства имен. Пространству имен назначается префикс пространства имен – он указывается после атрибута **xmlns** и отделяется двоеточием, а затем после знака равенства записывается адрес, однозначно идентифицирующий пространство имен. В приведенном примере образовано два пространства имен с префиксами **x** и **y**. Префиксы используются в тегах, указывая, к какому пространству имен относится данный тег.

Пример 2.5.3. Пространства имен **html** и **journal**.

```
<?xml version="1.0" encoding="windows-1251"?>
<!-- Пространства имен html и journal -->
<html xmlns:h="http://www.kcn.ru/three"
    xmlns:j="http://www.kcn.ru/four">
<h:head>
 <j:jrntitle>Lobachevsky&apos;Journal</j:jrntitle>
</h:head>
<h:body>
<j:jrntitle>Lobachevsky&apos;Journal</j:jrntitle>
<j:url>ljm.ksu.ru</j:url>
```

```
<j:articles>
<j:article ID="1">
<j:title>MathML and TeX</j:title>
</j:article>
<j:article ID="2">
<j:title>MathML and RDF</j:title>
</j:article>
</j:articles>
</h:body>
</html>
```

2.6. XSL-преобразование XML-документов

Как видно из приведенных примеров, язык XML совершенно не «заботится» о представлении информации, поскольку он разработан для других целей. Для отображения информации в «привычном» виде, как это делалось с помощью языка HTML, необходимо использовать другие технологии.

Универсальным механизмом управления отображением XML-документов является расширяемый язык таблиц стилей XSL (eXtensible Stylesheet Language). С помощью XSL можно трансформировать XML документ в любой формат, например HTML, WML, RTF. PDF, SQL и, в частности, в XML. На языке XSL можно описать, как будет оформлен итоговый документ, где и как должны располагаться данные.

Спецификация XSL [17] состоит из двух частей: XSL-T (XSL Transformations) – язык для преобразования XML-документов и XSL-FO (XSL Formatting Objects) – язык для вёрстки XML. Язык XSLT используется для преобразования XML-документа в документ, предназначенный для отображения браузером. Технически это осуществляется с помощью применения к исходному XML-документу *таблицы стилей* XSLT, состоящей из набора *шаблонов*. XSL-FO является языком разметки, который предназначен для описания внешнего вида (макета) документа. Мы не будем здесь рассматривать XSL-FO, отметим только, что схематически использование этого языка можно описать следующим образом. Исходным является документ на любом языке, являющимся подмножеством XML, например, XHTML или DocBook. Затем применяется XSLT-преобразование, подходящее к необходимому типу документа, и получается XSL-FO-документ. Этот документ передается приложению (FO-процессору), который конвертирует XSL-FO-документ в какой-либо читаемый и/или печатаемый формат, например, PDF, PS, RTF, или выводит его на монитор.

Перейдем к описанию XSLT и начнем с примеров. Создадим таблицу стилей для отображения файла **example2.xml**, приведенного в примере 2.3.2, и

сохраним ее в XSL-файле.

Пример 2.6.1. XSL-файл (one.xsl) для XML-файла из примера 2.3.2.

```
<xsl:stylesheet version="1.0"
 xmlns:xsl=" http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<h2>
 <xsl:value-of select="//jrntitle"/>
</h2>
<hr/>
<P>
<I><xsl:value-of select="//contacts/address"/></I>
</P>
<P>
<xsl:value-of select="//contacts/url"/>
</P>
</xsl:template>
</xsl:stylesheet>
```

Теперь в файл example2.xml добавим следующую ссылку на файл one.xsl:

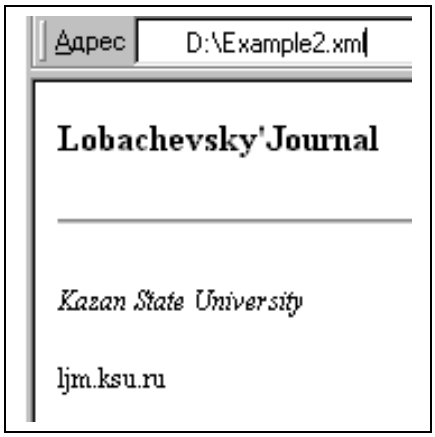
<?xml-stylesheet type='text/xsl' href='one.xsl'?>

Пример 2.6.2. Листинг файла example2.xml. Добавлена ссылка на XSL-файл.

```
<?xml version="1.0" encoding="windows-1251"?>
<?xml-stylesheet type='text/xsl' href='one.xsl'?>
<journal>
<jrntitle>Lobachevsky&amp;apos;Journal</jrntitle>
<contacts>
<address>Kazan State University</address>
<url>ljm.ksu.ru</url>
</contacts>
<articles>
<article ID="1">
<title>MathML and TeX</title>
<author>M. Malakhaltsev</author>
</article>
<article ID="2">
<title>MathML and RDF</title>
<author>E. Lipachev</author>
</article>
</articles>
</journal>
```

Если мы теперь откроем файл **example2.xml** в браузере Internet Explorer, то по-

лучим «привычную» форму представления документа:



На этом примере поясним принципы работы XSL-преобразований. Преобразование исходного XML-документа (в нашем случае, **example2.xml**) с помощью таблицы стилей (в нашем случае, **one.xsl**) осуществляется программным модулем, который называется XSLT-процессор. Большинство современных браузеров (например, Internet Explorer, Mozilla, Firefox) имеют встроенные XSLT-процессоры, позволяющие «на лету» осуществлять XSL-преобразование, то есть производить XSL-преобразование исходного XML-документа в соответствии с заданной таблицей стилей и передавать полученный документ другому модулю браузера для вывода на экран. В результате пользователь, открывая в браузере XML-документ, содержащий ссылку на таблицу стилей, видит документ, определяемый этой таблицей стилей. Таким образом, один и тот же XML-документ может быть показан в браузере различными способами (см. примеры ниже).

Отметим, что отображение информации в браузере – только одна из задач, для решения которых может быть применено XSL-преобразование XML-файлов. XSLT является мощным средством обработки информации, структурированной с помощью XML. Поэтому XSLT-процессоры включены в интерпретаторы языков, применяемых в веб-программировании (например, PHP, Java, JavaScript). В Linux XSLT-процессор включен в библиотеку libxml и может быть вызван в консоли командой xsltproc.

Кратко опишем алгоритм работы XSL-преобразования (детальное описание см. в [18], [19]).

Имеются исходный XML-файл (исходное дерево) и таблица стилей, содержащая набор правил шаблона. Правило шаблона состоит из двух частей: это образец, который сопоставляется с узлами в исходном дереве, и шаблон, который может быть обработан для формирования фрагмента в конечном дереве. В ходе XSL-преобразования образец сравнивается с элементами исходного дерева, а шаблон используется для создания частей конечного дерева. Отметим, что

структура конечного дерева может полностью отличаться от структуры исходного дерева. В ходе построения конечного дерева элементы исходного дерева могут подвергаться фильтрации и переупорядочению, также может быть добавлена новая структура.

Чтобы в конечном дереве построить фрагмент, обрабатывается перечень исходных узлов. Обработка узла осуществляется путем нахождения всех правил шаблона, чей образец соответствуют этому узлу, выбора среди них самого «лучшего» (правила, по которым выбирается лучший шаблон, см. в [18], [19]) и последующего применения к узлу инструкций выбранного шаблона. Этот процесс продолжается до тех пор, пока для обработки можно найти новые исходные узлы.

Посмотрим, как этот алгоритм работает в нашем примере. Начнем с таблицы стилей **one.xsl**. Таблица стилей является XML-файлом с корневым элементом

```
<xsl:stylesheet version="1.0"   xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
```

Его атрибуты – номер версии и ссылка на пространство имен (**xmlns:xsl**) – являются обязательными, а ссылка на **http://www.w3.org/1999/XSL/Transform** – стандартной.

Как было указано, таблица стилей состоит из правил шаблона. Эти правила задаются элементами **<xsl:template>**. При этом образец для выбора узлов XML-дерева, к которому будет применяться шаблон, задается атрибутом **match**, а шаблон является значением этого элемента. Таблица стилей **one.xsl** содержит одно правило шаблона

```
<xsl:template match="/">
<h2>
 <xsl:value-of select="//jrntitle"/>
</h2>
<hr/>
<P>
<I><xsl:value-of select="//contacts/address"/></I>
</P>
<P>
<xsl:value-of select="//contacts/url"/>
</P>
</xsl:template>
```

Образцом в этом правиле является **"/"**, а самим шаблоном – значение элемента **<xsl:template>**, то есть

```
<h2>
```

```
 <xsl:value-of select="//jrntitle"/>
</h2>
<hr/>
<P>
<I><xsl:value-of select="//contacts/address"/></I>
</P>
<P>
<xsl:value-of select="//contacts/url"/>
</P>
```

Образец шаблона является выражением, написанным на языке Xpath, который предназначен для адресации узлов (и подмножеств узлов) XML-дерева. В данном руководстве нет возможности подробно описать синтаксис и применения языка Xpath, поэтому мы вынуждены ограничиться тем, что приведем несколько простых примеров выражений. Подробно об этом языке можно узнать из рекомендаций W3C [20] (также см. руководства [21], [22], [23]). Отметим, что в 2007 году завершилась разработка версии 2.0 языка XPath, который теперь является составной частью языка XQuery.

Примеры выражений на языке Xpath:

Выражение	**Результат**
paper	Выбирает всех прямых потомков узла paper
/paper	Выбирает элемент paper, являющийся прямым потомком корневого узла **Замечание:** Если выражение начинается с /, то оно описывает абсолютный путь к элементу
paper/title	Выбирает все узлы title, являющиеся прямыми потомками paper
//paper	Выбирает все узлы paper, независимо от того, где этот элемент находится в дереве
journal//author	Выбирает все элементы author являющиеся потомками элементов journal
//@lang	Выбирает все атрибуты с именем lang

В нашем примере значение **"/"** атрибута **match** определяет, что шаблон будет применяться ко всем элементам XML-дерева.

Теперь перейдем к инструкциям шаблона. Вначале идет текстовый узел **<h2>**. Каждый текстовый узел в шаблоне создаст в конечном дереве текстовый узел с тем же самым строковым значением, то есть в результирующее дерево будет записано **<h2>**.

Далее идет элемент**<xsl:value-of select="//jrntitle"/>**. Этот элемент позволяет получить значение элемента исходного XML-документа, определяемого значением атрибута **select**, то есть **"//jrntitle"** (отметим, что значениями этого атрибута являются XPath-выражения). В нашем примере в результирующее дерево будет записано значение элемента

<jrntitle> Lobachevsky'Journal</jrntitle>

исходного XML-дерева из файла Example2.xml, то есть

Lobachevsky'Journal

Применение остальных инструкций шаблона происходит аналогично и в результате получается документ

```
<?xml version="1.0"?>
<h2>Lobachevsky'Journal</h2>
<hr/>
<P><I>Kazan State University</I></P>
<P>ljm.ksu.ru</P>
```

который отображается браузером вышеуказанным способом.

Теперь покажем как, изменив стилевую таблицу, можно из того же самого файла **Example2.xml** получить файл, содержащий список авторов журнала. Возьмем вначале стилевую таблицу

```
<xsl:stylesheet version="1.0"  xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="//article">
<xsl:value-of select="author"/>
</xsl:template>
</xsl:stylesheet>
```

Эта таблица содержит одно правило шаблона, образцом является //**article**, поэтому можно предположить, что результирующий документ будет представлять собой список авторов:

```
M. Malakhaltsev
E. Lipachev
```

На самом деле, мы получим следующий документ:

```
<?xml version="1.0"?>
Lobachevsky'Journal
Kazan State University
ljm.ksu.ru
M. Malakhaltsev
E. Lipachev
```

Дело в том, что для узлов, которые не удовлетворяют ни одному образцу, применяется шаблон по умолчанию, который в данном случае просто выводит содержимое соответствующих элементов исходного XML-файла. Если же применить следующую таблицу стилей

```
<xsl:stylesheet version="1.0"  xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
```

```
<xsl:template match="/">
</xsl:template>
<xsl:template match="//article">
<xsl:value-of select="author"/>
</xsl:template>
</xsl:stylesheet>
```

которая содержит два правила шаблона (шаблон первого правила является пустым для того, чтобы подавить вывод ненужной информации), для элементов **article** возникнет конфликт правил шаблона (этот элемент удовлетворяет двум образцам). Этот конфликт разрешается в пользу первого правила шаблона – «ничего не делать» (подробно о разрешении конфликтов правил шаблона и о том, как выйти из этой ситуации, см. [18], [19]).

Чтобы все же получить список авторов, можно попытаться применить следующее правило шаблона:

```
<xsl:stylesheet version="1.0"  xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="/">
<xsl:value-of select="//article/author"/>
</xsl:template>
</xsl:stylesheet>
```

Однако результатом будет документ, содержащий только первого автора

```
M. Malakhaltsev
```

Причина в том, что к корневому элементу применяется шаблон, который указывает, что надо вывести содержимое элемента, определяемого выражением **//article/author**. Однако таких элементов в данном случае два и нет указаний, какой из них надо выбрать. Поэтому выбирается первый.

Для того чтобы вывести список всех авторов, применяется элемент **<xsl:for-each>**, дающий инструкцию осуществить обработку каждого элемента из набора элементов, определенного атрибутом **select**, значением которого является XPath-выражение. В данном случае стилевая таблица будет иметь вид:

```
<xsl:stylesheet version="1.0"  xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="/">
<xsl:for-each select="//article">
<p>
<xsl:value-of select="author"/>
</p>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

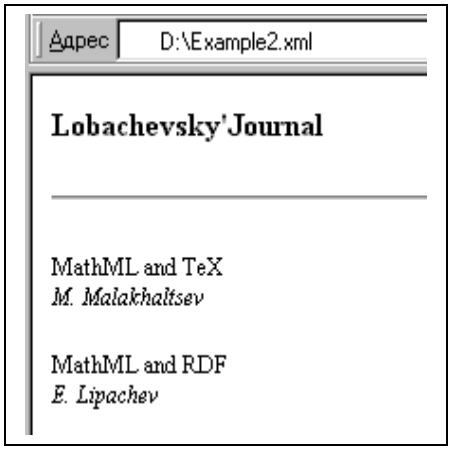
При применении этого шаблона последовательно обрабатываются ветви XML-дерева, начинающиеся с элементов **article** и для каждой из них выводится содержимое элемента **author**.

Приведем еще несколько примеров XSLT-преобразований для вывода информации, содержащейся в файле example2.xml в разных видах.

Пример 2.6.3. Ещё одна версия файла one.xsl.

```
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
<h2>
 <xsl:value-of select="journal/jrntitle"/>
</h2>
<hr/>
<xsl:for-each select="journal/articles/article">
<P>
<xsl:value-of select="title"/><br/>
<I><xsl:value-of select="author"/></I>
</P>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

При запуске XML-файла example2.xml из примера 2.6.2 результат будет иной:



Если в файле one.xsl изменить тег **<xsl:for-each** … на

<xsl:for-each select="journal/articles/article" order-by="author">,

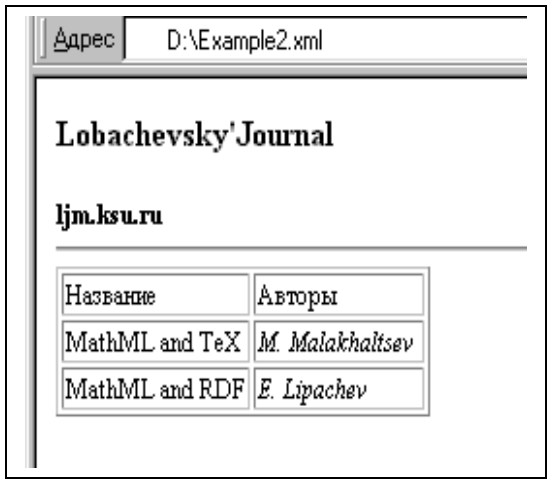
то перед выводом статьи будут упорядочены по авторам. Критерий сортировки задаётся атрибутом **order-by,** при этом символ **";"** используется как разделитель, если критериев несколько.

Пример 2.6.4. Следующий стилевой файл используется для отображения XML-файла в виде таблицы. Обращаем внимание на русские названия столбцов таблицы. Для этого в XSL-файле с помощью атрибута **encoding** указана кодировка символов.

```
<?xml version="1.0" encoding="windows-1251"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
```

```
<h2>
  <xsl:value-of select="journal/jrntitle"/>
</h2>
<b><xsl:value-of select="journal/contacts/url"/> </b>
<hr/>
<table border="1">
<tr>
 <td> Название </td><td> Авторы </td>
</tr>
<xsl:for-each select="journal/articles/article">
<tr>
 <td><xsl:value-of select="title"/></td>
 <td><I><xsl:value-of select="author"/></I></td>
</tr>
</xsl:for-each>
</table>
</xsl:template>
</xsl:stylesheet>
```

Если этот файл сохранить как **one.xsl**, то при запуске файла **example2.xml,** код которого приведен ранее, получим



Язык XSLT имеет все управляющие конструкции языков программирования, например, как мы видели, элемент <xsl:for-each> задает оператор цикла. В рамках настоящего руководства мы не будем описывать все возможности языка XSLT (они изложены в http://www.w3.org/TR/1999/REC-xslt-19991116; http://www.rol.ru/news/it/helpdesk/xslt01.htm), а приведем пример условного оператора. Рассмотрим следующую таблицу стилей:

```
<xsl:stylesheet version="1.0" xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="/">
<xsl:for-each select="//article">
<p>
```

```
<xsl:if test="@ID=1">
<xsl:value-of select="author"/>
</xsl:if>
</p>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

В результате применения этой стилевой таблицы к файлу example2.xml получим

```
<?xml version="1.0"?>
<p>M. Malakhaltsev</p>
<p></p>
```

При обходе ветвей, начинающихся с элемента **article**, проверяется значение атрибута **Id** этого элемента, и, если он равен 1, то выводится значение элемента **author**. Элемент xsl:if имеет атрибут test, который определяет некое выражение. Содержимое элемента является шаблоном. Указанное выражение обрабатывается, а полученный объект преобразуется в булево значение как при вызове функции boolean. Если результатом является true, то подставляется шаблон, имеющийся в выражении. В противном случае не создается ничего.

Приведенные примеры показывают, что язык XSLT является удобным средством обработки информации, структурированной с помощью XML; с его помощью можно решать достаточно широкий круг задач, связанных с представлением информации в различных формах.

3. Разметка математических текстов по технологии MathML

3.1. Вводный пример

Допустим, что нам надо осуществить MathML-разметку для отображения следующего текста:

Quadratic equation

Корни квадратного уравнения $ax^2+bx+c=0$

имеют вид

$$x_{1,2}=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

Код для отображения этого фрагмента можем подготовить в стандартном блокноте Windows. Откроем блокнот и наберем:

```
<?xml version="1.0" encoding=”windows-1251”?>
```

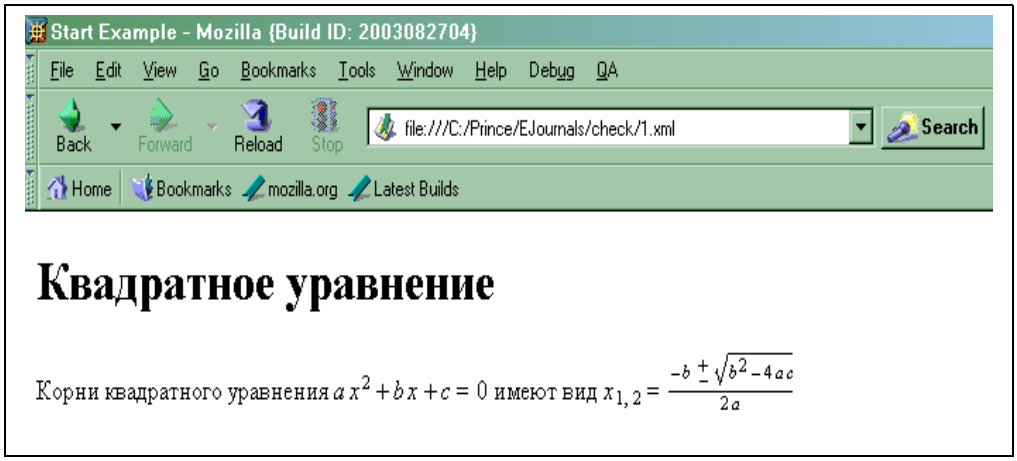
```
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
<title>Start Example</title>
</head>
 <body>
  <h1>Quadratic equation</h1>
 Корни квадратного уравнения
<math xmlns="http://www.w3.org/1998/Math/MathML">
    <mi>a</mi>
   <msup>
     <mi>x</mi>
     <mn>2</mn>
  </msup>
  <mo>+</mo>
  <mi>b</mi>
  <mi>x</mi>
   <mo>+</mo>
  <mi>c</mi>
  <mo>=</mo>
  <mn>0</mn>
 </math>
```

имеют вид

```
 <math xmlns="http://www.w3.org/1998/Math/MathML">
   <msub>      <mi>x</mi>
     <mrow>
      <mn>1</mn>
     <mo>,</mo>
     <mn>2</mn>
     </mrow>
   </msub>
   <mo>=</mo>
   <mfrac>
    <mrow>
       <mo>-</mo>
       <mi>b</mi>
       <mfrac linethickness="0">
         <mo>+</mo>
         <mo>-</mo>
       </mfrac>
       <msqrt>
         <msup>
           <mi>b</mi>
```

```
            <mn>2</mn>
          </msup>
          <mo>-</mo>
          <mn>4</mn>
          <mi>a</mi>
          <mi>c</mi>
        </msqrt>
      </mrow>
      <mrow>
        <mn>2</mn>
        <mi>a</mi>
      </mrow>
    </mfrac>
  </math>
 </body>
</html>
```

Теперь сохраним файл под именем **example1.xml**. Откроем его в браузере Mozilla, мы должны увидеть картинку



Теперь попытаемся разобраться в тексте созданного нами файла. Первые две строки указывают, что данный документ представляет код XHTML. Напомним, что первая строка отмечает, что это XML-документ, а вторая определяет пространство имен XHTML как **http://www.w3.org/1999/xhtml.** Хотя эта строка выглядит как ссылка, она является лишь уникальным идентификатором пространства имен, который распознается браузером, и, следовательно, не требует соединения с интернетом. Отметим, что пространство имен используется по умолчанию (т. к. нет префикса после **xmlns**), поэтому перед тегами не указан префикс, задающий пространство имен (подробно о пространстве имен см. выше раздел 2.6).

Для разметки текста используются стандартные теги HTML, а для пред-

ставления формул – теги MathML. Блоки с кодом MathML заключатся в теги

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
…….
</math>
```

Открывающий тег **<math ….>** определяет ссылку на пространство имен MathML. В примере необходимо отобразить две формулы, поэтому он содержит два блока **$$.**

Выражения MathML строятся в виде дерева, каждый узел которого является элементом MathML (например, числом или операцией). Все элементы MathML разделены на три группы: *элементы представления* (presentation elements), *элементы содержания* (content elements) и *интерфейсные элементы* (interface elements). Одно и то же выражение может быть размечено в виде «разметки представления» (presentation markup) и «разметки содержания» (content markup). В нашем примере используется разметка представления. *Так как разметка содержания в настоящий момент не поддерживается браузерами, мы ее не рассматриваем.* В рекомендации MathML [24] элементы представления называются также *токенами представления* (*presentation token elements*).

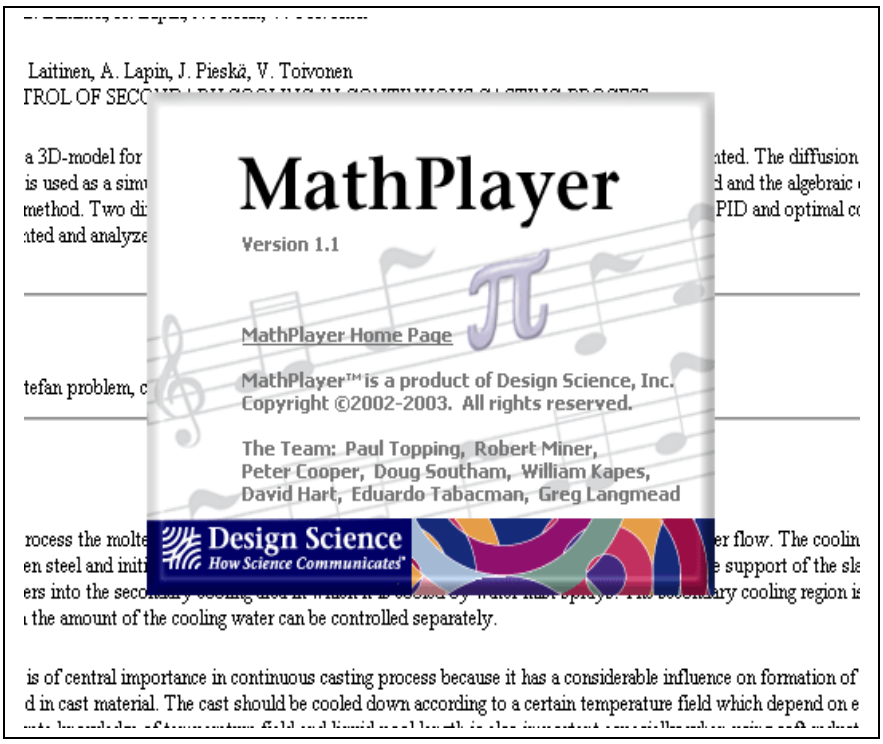
Элементы представления **mi, mn** и **mo** используются чаще всего – с их помощью производится разметка для идентификаторов (например, переменных), чисел и операций. В примере имеются элементы **msup** и **msub,** которые используются для отображения верхних и нижних индексов. Во второй формуле нижний индекс двойной, поэтому потребовалась группировка элементов, представляющих индекс. Для группировки использовали элемент **mrow**, который предназначен для обозначения горизонтального ряда частей выражения. Элемент **msqrt** отвечает за представление квадратного корня, а элемент **mfrac** – за представление дроби. Подробно теги MathML описаны ниже.

3.2. Особенности отображения в браузерах

В настоящее время в основном используются браузеры Internet Explorer, Firefox, Mozilla. Браузеры Firefox и Mozilla отображают MathML одинаково, поэтому мы объясним различие в подготовке MathML-разметки для Internet Explorer и Mozilla.

3.2.1. Особенности отображения в Internet Explorer

Для просмотра MathML в Internet Explorer необходимо «скачать» пакет MathPlayer (например, с сайта http://www.dessci.com/en/). Установка не вызовет затруднений. В каждом сеансе работы с Internet Explorer при открытии MathML-файла загружается MathPlayer, сообщая о себе выводом окна



Просмотр XML-файла. Прежде надо скачать с сети файл **pmathml.xsl**. В начало файла помещаются строки

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="pmathml.xsl"?>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html" />
</head>
<body>
```

Здесь предполагается, что файл pmathml.xsl размещен в одном каталоге с создаваемым xml–файлом, в противном случае требуется указать полный путь к нему. Каждый фрагмент MathML заключается в теги:

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
</math>
```

В конце файла не забудьте поставить закрывающие теги

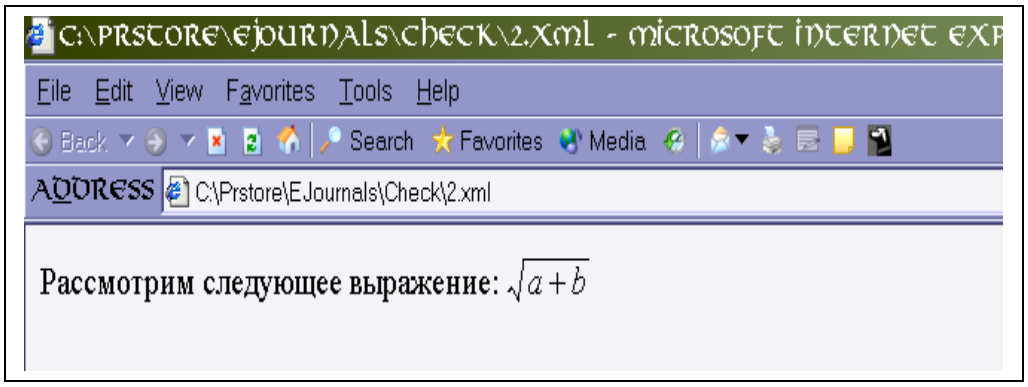
```
</body>
</html>
```

Пример XML-файла для браузера Internet Explorer. Наберите в Блокноте следующий текст:

```
<?xml version="1.0" encoding="windows-1251"?>
<?xml-stylesheet type="text/xsl" href="pmathml.xsl"?>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html" />
```

```
</head>
<body>
<p> Рассмотрим следующее выражение:
<math xmlns="http://www.w3.org/1998/Math/MathML">
 <msqrt>
 <mi>a</mi>
 <mo>+</mo>
 <mi>b</mi>
 </msqrt>
</math>
</p>
</body>
</html>
```

Сохраните этот файл как **2.xml**. Открыв этот файл в браузере Internet Explorer, вы должны увидеть следующую картинку:



Просмотр HTML-файла. Приведем простейший пример html-файла, содержащего MathML-код, использующий MathPlayer для отображения:

```
<HTML XMLNS:m="http://www.w3.org/1998/Math/MathML">
<head>
  <OBJECT ID=example1
CLASSID="clsid:32F66A20-7614-11D4-BD11-00104BD3F987">
  </OBJECT>
  <?IMPORT NAMESPACE="M" IMPLEMENTATION="#example1">
</head>

<body>
<p> Рассмотрим следующее выражение
 <m:math>
  <m:msqrt>
      <m:mi>a</m:mi>
      <m:mo>+</m:mo>
      <m:mi>b</m:mi>
   </m:msqrt>
```

```
  </m:math>
</p>
</body>
</html>
```

MathPlayer подключается при помощи тега **OBJECT**. В нем указывается атрибут ID, значение которого можно изменять (это же значение используется в теге **IMPORT**).

В html-файл можно добавить скрипт, который проверяет версию браузера. Примерно такой же код получается при конвертации TeX-файлов в MathML программой TeX4ht (см. раздел 6.5).

Начало файла содержит следующие обязательные строки:

```
<html xmlns:m="http://www.w3.org/1998/Math/MathML"
  xmlns="http://www.w3.org/1999/xhtml" >

<head>
<object
id="MathPlayer"
classid="clsid:32F66A20-7614-11D4-BD11-00104BD3F987">
</object>
<?import namespace="m" implementation="#MathPlayer" ?>
</head>
```

В начале раздела **<body>** надо вызвать MathPlayer:

```
<body>

<!--l. 31--><p class="noindent"><script language = "javascript">
<!--
if( navigator.appName=="Microsoft Internet Explorer"
&& navigator.platform=="Win32"
&& parseFloat(navigator.appVersion.substr(
navigator.appVersion.indexOf("MSIE ")+5))>="5.5"
){
try {
var oMP = new ActiveXObject("MathPlayer.Factory.1");
}
catch(e) { alert("Can't find Design Science's MathPalyer" +
"(http://www.dessci.com/webmath/mathplayer)");}
} else {
alert("Requires MSIE version 5.5 or later");
}

-->
```

```
</script>
</p>
```

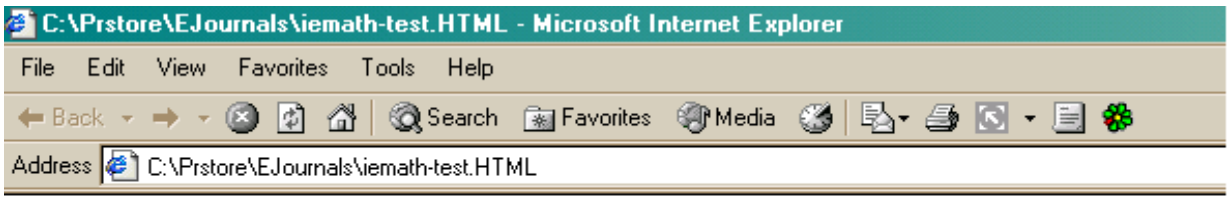
Синтаксис MathML-разметки также отличается от xml-варианта. Именно, каждый тег начинается с **<m:**, например:

```
<m:math>
<m:mi>T</m:mi>
<m:mo>=</m:mo>
<m:mi>A</m:mi>
</m:math>
```

Таким образом, формально это выглядит как использование CSS (Cascade Style Sheets), однако фактически соответствующий CSS-лист встроен в Math-Player. В частности, можно использовать атрибут **class**, например,

```
<m:math  display="inline">
<m:mrow>
  <m:mo class="MathClass-open">[</m:mo>
   <m:mrow>
    <m: msub>
     <m:mrow><m:mi>W</m:mi></m:mrow>
     <m:mrow><m:mn>0</m:mn></m:mrow>
    </m:msub>
    <m:mo class="MathClass-punc">,</m:mo>
    <m:msub>
       <m:mrow><m:mi>L</m:mi></m:mrow>
       <m:mrow><m:mi>X</m:mi></m:mrow>
    </m:msub>
   </m:mrow>
 <m:mo class="MathClass-close">]</m:mo>
</m:mrow>
<m:mo class="MathClass-bin">&#x00D7;</m:mo>
<m:mi> Y </m:mi>
<m:mo class="MathClass-rel">=</m:mo>
<m:mrow>
<m:mi> Y </m:mi>
</m:math>
```

При просмотре этот фрагмент выглядит так:



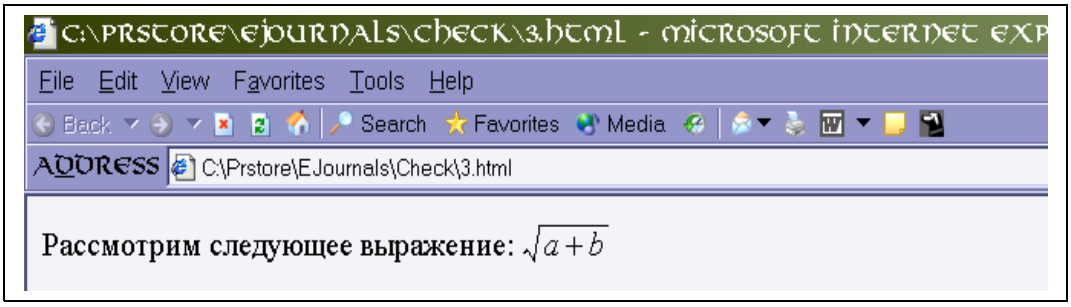
$$\left[W_0, L_X\right] \times Y = Y$$

Пример HTML-файла для браузера Internet Explorer. Наберите в Блокноте этот текст:

```
<html xmlns:m="http://www.w3.org/1998/Math/MathML"
xmlns="http://www.w3.org/1999/xhtml">
<head>
<object id="MathPlayer"
classid="clsid:32F66A20-7614-11D4-BD11-00104BD3F987">
</object>
<?import namespace="m" implementation="#MathPlayer" ?>
<!--http://www.dessci.com/webmath/mathplayer/-->
</head>
<body>
<!--l. 31--><p class="noindent">
<script language  ="javascript">
<!--
if( navigator.appName=="Microsoft Internet Explorer"
&& navigator.platform=="Win32"
&& parseFloat(navigator.appVersion.substr(
navigator.appVersion.indexOf("MSIE")+5))>="5.5"
){
try {
var oMP = new ActiveXObject("MathPlayer.Factory.1");
}
catch(e) {
 alert("Can't find Design Science's Math-Palyer" +
"(http://www.dessci.com/webmath/mathplayer)");}
}
else {
alert("Requires MSIE version 5.5 or later");
}
-->
</script>
</p>
<p>
Рассмотрим следующее выражение:
```

```
<m:math>
 <m:msqrt> <m:mrow> <m:mi>a</m:mi>
 <m:mo>+</m:mo>
 <m:mi>b</m:mi>
 <m:mrow>
 </m:msqrt>
</m:math>
</p>
</body>
</html>
```

Сохраните файл как 3.html. Вы должны увидеть следующую картинку:



Пакет TechExplorer. Кроме пакета (точнее, плагина) MathPlayer, для отображения MathML-документов в браузере Internet Explorer можно использовать другой плагин. Он называется TechExplorer, информация о нем находится на сайте http://www.integretechpub.com/techexplorer/. Подробности использования программы TechExplorer можно найти в [18].

3.2.2. Особенности отображения в браузере Mozilla

Браузер Mozilla имеет встроенную поддержку MathML для формата XML. Файл, в котором имеются MathML-включения, надо начинать со строк

```
<?xml version="1.0"?>
<html xmlns="http://www.w3.org/1999/xhtml">
```

Каждый фрагмент MathML заключается в теги:

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
…….
</math>
```

В конце файла ставятся закрывающие теги

```
</body>
</html>
```

Так, уже рассмотренный пример для браузера Internet Explorer будет для браузера Mozilla выглядеть следующим образом:

```
<?xml version="1.0"?>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<body>
<p> Рассмотрим следующее выражение
<math xmlns="http://www.w3.org/1998/Math/MathML">
 <msqrt> <mrow> <mi>a</mi>
 <mo>+</mo>
 <mi>b</mi>
 </mrow>
 </msqrt>
</math>
</p>
</body>
</html>
```

Открыв этот файл в браузере Mozilla, вы увидите такую же картинку, какая была в браузере Internet Explorer.

4. Синтаксис языка

Из примеров, приведенных в предыдущих разделах, мы видим, что разметка MathML очень громоздкая, поэтому вручную ввести реальный текст невозможно (впрочем, если читатель привык к TeX-набору, он без особого труда справится с MathML-набором; надо лишь использовать редактор с автозаменой). В дальнейшем мы описываем программы, которые автоматически генерируют MathML-код (см. главу 6 настоящего руководства). Тем не менее, надо иметь представление о синтаксисе языка, так как часто приходится поправлять сгенерированный код вручную.

Если проанализировать уже приведенные примеры, то увидим, что каждый токен (элемент) формулы заключается в тег с атрибутами, указывающими его специализацию. Например, число заключается в тег **<mn> </mn>,** операция – в **<mo> </mo>,** символ – в **<mi> </mi>.** При этом одни теги могут быть вложены друг в друга. Таким образом, выражения MathML можно представить в виде деревьев, где каждый узел соответствует элементу MathML, ветвь под «родительским» узлом соответствует «дочерним» узлам и листья дерева соответствуют атомарным элементам нотации или содержания, таким, как числа, символы и т. д. Все элементы MathML делятся на три группы: элементы представления, элементы содержания и интерфейсные элементы. Элементы представления описывают визуально ориентированную двухмерную структуру математической нотации. Типичным примером является элемент **mrow,** который обычно применяется для обозначения горизонтального ряда частей выражения, и элемент **msup,** который отмечает верхний индекс. Как правило, каждый элемент пред-

ставления соответствует одному типу нотационной схемы, такой, как ряд, верхний индекс, нижний индекс и так далее. Любая формула состоит из частей, которые могут состоять из простейших элементов, таких, как цифры, буквы или другие символы.

Разметка MathML состоит из приблизительно 30 элементов, которые имеют более 50 атрибутов. Элементы представления делятся на два класса. *Токены (token elements)* представляют индивидуальные символы, названия, числа, обозначения и т. д. В основном в качестве содержания токены могут иметь только символы. Единственными исключениями являются элемент вертикального выравнивания malignmark, mglyph и ссылки на сущности (entity references). *Элементы схемы (layout schemata)* формируют выражения из частей и могут иметь только элементы в качестве содержания (исключая пробелы, которые они игнорируют). Также имеется несколько пустых элементов, используемых только вместе с определенным элементом схемы.

Все индивидуальные «символы» в математическом выражении должны быть представлены токенами MathML. Основные типы токенов MathML-идентификаторы (т. е. переменные или имена функций), числа и операторы (включая различные ограничители, такие, как круглые скобки, и разделители, такие, как запятые). Имеются также токены для представления текста или пробелов, что имеет больше эстетическое, чем математическое значение, и для представления «строчных констант» (string literals) для совместимости с системами компьютерной алгебры. Надо заметить, что хотя токен представляет одиночную смысловую единицу (имя, число, знак, математический символ и т. д.), такая единица может состоять более чем из одной литеры (character). Например, **sin** и **24** представлены одиночными токенами **<mi>sin</mi>** и **<mn>24</mn>** соответственно.

Далее мы приводим основные элементы MathML (более подробное изложение с перечислением всех тегов и многочисленных атрибутов можно найти в [24]).

4.1. Токены

Тег	**Определение**	**Пример**
mi	идентификатор (identifier)	**<mi> a </mi>**
mn	число (number)	**<mn> 1324 </mn>**
mo	оператор (operator), ограничитель (fence) или разделитель (separator)	**<mo>+</mo>**
mtext	текст (text)	**<mtext> Это пример </mtext>**

ms	текстовая строка (string literal)	**<ms> HHHHHHH</ms>**

Применение основных токенов ясно из вышеприведенных примеров.

4.2. Основные элементы

Тег	Определение	Пример	Результат
mrow	группирует любое количество подвыражений в строку	см. вводный пример	
mfrac	формирует дробь из двух подвыражений	**<mfrac> <mi>a</mi> <mi>b</mi> </mfrac>**	$\frac{a}{b}$
msqrt	формирует квадратный корень (радикал без индекса)	**<math> <msqrt> <mi>a</mi> <mo>+</mo> <mi>b</mi> </msqrt> </math>**	$\sqrt{a+b}$
mroot	формирует радикал с определенным индексом	**<math> <mroot> <mrow> <mi>a</mi> <mo>+</mo> <mi>d</mi> </mrow> <mrow> <mi>b</mi> <mo>+</mo> <mi>c</mi> </mrow> </mroot> </math>**	$\sqrt[b+c]{a+d}$
mpadded	регулирует отступы вокруг содержимого	**<math> <mpadded width="+5em"> <mi>a</mi> <mo>+</mo> <mi>b</mi> </mpadded> <mo>-</mo> <mi>c</mi> </math**	$a+b$
mphantom	делает содержимое не-	**<math>**	

	видимым, но сохраняет его размер	**<mi> x </mi> <mphantom> <mo> + </mo> <mi> y </mi> </mphantom> <mo> + </mo> <mi> z </mi> </math>**	$x \quad +z$
mfenced	окружает содержимое парой скобок	**<math> <mfenced> <mi>x</mi> </mfenced> </math>**	(x)

4.3. Индексы

Тег	Определение	Пример	Результат
msub	добавляет нижний индекс к основанию	**<math> <msub> <mi>x</mi> <mn>1</mn> </msub> </math>**	x_1
msup	добавляет верхний индекс к основанию	**<math> <msup> <mi>x</mi> <mn>12</mn> </msup> </math>**	x^{12}
msubsup	добавляет верхний и нижний индексы к основанию	**<math> <msubsup> <mi>x</mi> <mn>z</mn> <mn>12</mn> </msubsup> </math>**	x_z^{12}
munder	добавляет символ под основанием	**<math> <munder> <mi>X</mi> <mi>h</mi> </munder> </math>**	$\underset{h}{X}$
mover	добавляет символ над основанием	**<math> <mover> <mi>X</mi> <mi>h</mi> </mover> </math>**	$\overset{h}{X}$
munderover	добавляет символы одновременно и над, и под основанием	**<m:math> <m:munderover> <m:mi>X</m:mi> <m:mi>h</m:mi>**	

		<m:mi>t</m:mi> **</m:munderover>** **</m:math>**	
mmultiscripts	добавляет мультиин-дексы к основанию	**<math>** **<mmultiscripts>** **<mi> R </mi>** **<mi> i </mi>** **<mi> j </mi>** **<mi> k </mi>** **</mmultiscripts>** **</math>**	

5. Более сложные примеры

В этом разделе мы приводим несколько примеров, построенных на основе элементов, описанных в предыдущем разделе. Мы не останавливаемся на используемых здесь атрибутах, так как эта техническая информация приведена в [21].

5.1. Формула сокращенного умножения

Код MathML:

```
<math display="inline">
  <msup>
 <mrow >
  <mrow>
   <mo class="MathClass-open">(</mo>
   <mrow>
    <mi>a</mi>
    <mo class="MathClass-bin">+</mo>
    <mi>b</mi>
   </mrow>
   <mo class="MathClass-close">)</mo>
  </mrow>
</mrow>
<mrow>
 <mn>2</mn>
</mrow>
        </msup>
        <mo class="MathClass-rel">=</mo>
        <msup>
        <mrow>
         <mi>a</mi>
        </mrow>
        <mrow>
```

```
        <mn>2</mn>
       </mrow>
      </msup>
      <mo class="MathClass-bin">+</mo>
      <mn>2</mn><mi>a</mi><mi>b</mi>
      <mo class="MathClass-bin">+</mo>
       <msup>
      <mrow>
      <mi>c</mi>
      </mrow>
      <mrow><mn>2</mn>
      </mrow>
      </msup>
  </math>
```

Результат:

$$(a+b)^2=a^2+2ab+c^2$$

5.2. Основное тригонометрическое тождество

Код MathML:

```
<math  display="inline">
<mo class="MathClass-op">sin</mo>
<msup>
 <mrow>
  <mrow>
   <mo class="MathClass-open">(</mo>
    <mrow>
     <mi >x</mi>
    </mrow>
 <mo class="MathClass-close">)
 </mo>
</mrow>
      </mrow>
      <mrow>
       <mn>2</mn>
      </mrow>
    </msup>
    <mo class="MathClass-bin">+</mo>
    <mo class="MathClass-op"> cos</mo>
    <msup>
     <mrow>
      <mrow>
```

```
        <mo class="MathClass-open">(</mo>
         <mrow>
          <mi>x</mi>
         </mrow>
         <mo class="MathClass-close">)</mo>
        </mrow>
       </mrow>
       <mrow>
       <mn>2</mn>
       </mrow>
      </msup>
      <mo class="MathClass-rel">=</mo>
      <mn>1</mn>
      </math>
```

Результат:

$$\sin(x)^2+\cos(x)^2=1$$

5.3. Формула с греческими буквами и символом частного дифференцирования

Для задания букв греческого алфавита и математических символов в MathML-коде используется юникод [19].

Код MathML:

```
<math   display="inline">
<msub>
 <mrow><mi>&#x2202;</mi></mrow>
  <mrow><mi>i</mi></mrow>
</msub>
<mrow>
  <mo class="MathClass-open">[</mo>
  <mrow>
    <mover accent="true">
     <mrow>  <mi>&#x03B1;</mi>   </mrow>
     <mo class="MathClass-op">&#x2192;</mo>
    </mover>
    <mo class="MathClass-punc">,</mo>
    <mover accent="true">
     <mrow><mi>&#x03B2;</mi></mrow>
     <mo class="MathClass-op">&#x2192;</mo>
    </mover>
   </mrow>
   <mo class="MathClass-close">]</mo>
```

```
  </mrow>
  <mo class="MathClass-rel">=</mo>
  <mover accent="true">
    <mrow><mi>&#x03B3;</mi></mrow>
    <mo class="MathClass-op">&#x2192;</mo>
  </mover>
</math>
```

Результат:

$$\partial_i[\vec{a}, \vec{\beta}] = \vec{\gamma}$$

5.4. Матрица

При написании MathML-кода матрицы используются следующие теги:

Тег	Определение
mtable	таблица или матрица
mtr	строка в таблице или матрице
mtd	одна ячейка в таблице или матрице

Код MathML:

```
<math  display="block">
<mfenced separators="" open="("  close=")" >
<mrow>
 <mtable>
   <mtr>
     <mtd> <mi>A</mi></mtd>
     <mtd> <mi>B</mi> </mtd>
   </mtr>
   <mtr>
     <mtd><mi>C</mi></mtd>
     <mtd><mi>D</mi></mtd>
   </mtr>
 </mtable>
</mrow>
</mfenced>
</math>
```

Результат:

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

5.5. Коммутативная диаграмма

Используя матрицы, можно построить коммутативные диаграммы. Отметим, что они лучше всего воспроизводятся с помощью MathPlayer.

MathML-код:

```
<math  display="block">
 <mtable >
  <mtr>
   <mtd>
    <mi>A</mi>
   </mtd>
   <mtd>
    <mover>
     <mrow>
     <mo class="MathClass-rel">&#x2192; </mo>
     </mrow>
     <mrow><mi>f</mi></mrow></mover>
   </mtd>
   <mtd>
     <mi>B</mi>
   </mtd>
  </mtr>
  <mtr>
   <mtd>
     <mi>&#x2193;</mi>
   </mtd>
   <mtd>
   </mtd>
   <mtd>
     <mi>&#x2191;</mi>
   </mtd>
  </mtr>
  <mtr>
  <mtd>
   <mi>C</mi>
  </mtd>
  <mtd>
   <munder>
    <mrow>
     <mo class="MathClass-rel"> &#x2192; </mo>
    </mrow>
     <mrow><mi>g</mi></mrow>
   </munder>
```

```
      </mtd>
      <mtd>
        <mi>D</mi>
      </mtd>
    </mtr>
  </mtable>
  </math>
```

Результат:

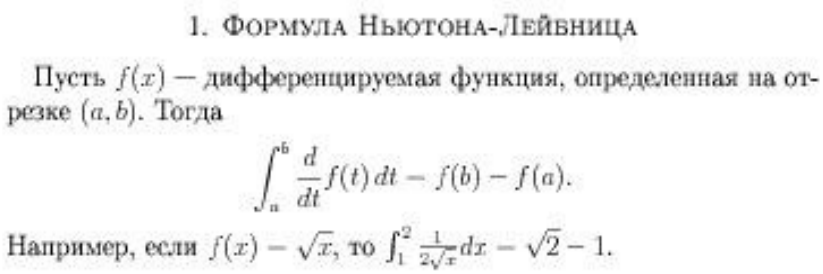
$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ \downarrow & & \uparrow \\ C & \xrightarrow[g]{} & D \end{array}$$

6. Инструменты для работы с MathML

В этом параграфе рассмотрим создание MathML-файла, содержащего один и тот же математический текст, с помощью различных программных средств различных редакторах. В качестве примера возьмем следующий текст, полученный с помощью издательской системы LaTeX:

```
\documentclass[12pt]{amsart}
\usepackage[russian]{babel}
\usepackage[utf8]{inputenc}
\begin{document}
\section{Формула Ньютона - Лейбница}
Пусть $f(x)$~--- дифференцируемая функция, определенная на отрезке
$(a,b)$. Тогда
$$
\int^b_a \frac{d}{dt}f(t)\,dt = f(b)-f(a).
$$
Например, если $f(x)=\sqrt{x}$, то
$\int^2_1 \frac{1}{2\sqrt{x}}dx = \sqrt{2}-1$.
\end{document}
```

Сохраним этот текст в файл example.tex. После компиляции получим example.dvi, который просматривается следующим образом:



1. Формула Ньютона-Лейбница

Пусть $f(x)$ — дифференцируемая функция, определенная на отрезке (a, b). Тогда

$$\int_a^b \frac{d}{dt} f(t)\, dt - f(b) - f(a).$$

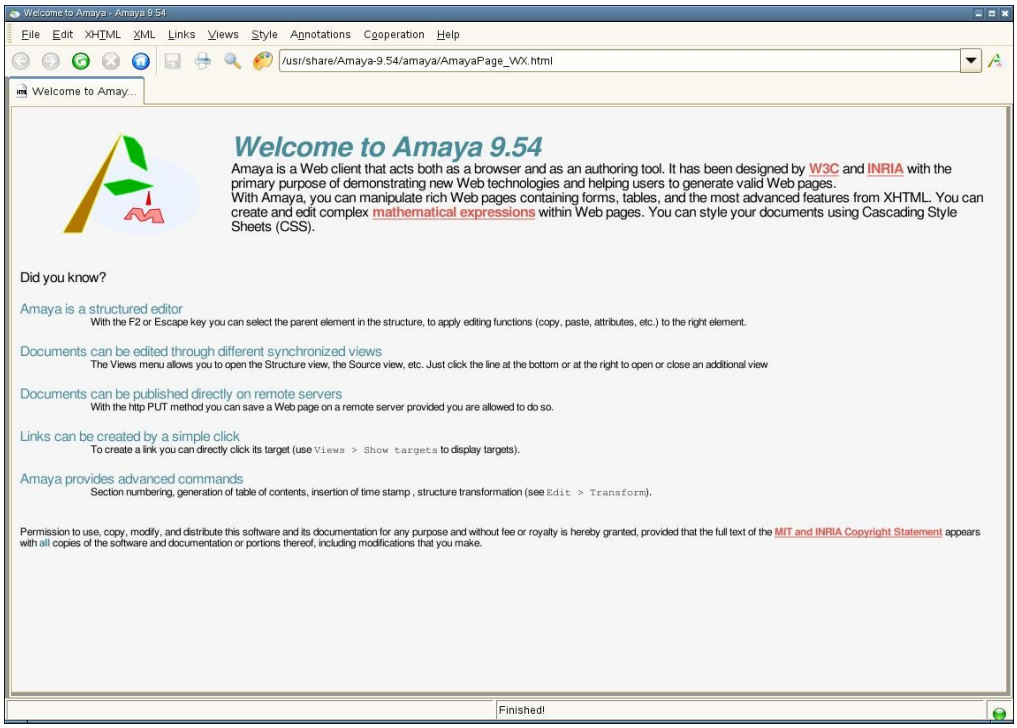
Например, если $f(x) - \sqrt{x}$, то $\int_1^2 \frac{1}{2\sqrt{x}} dx - \sqrt{2} - 1$.

Все рассматриваемые нами программные средства работают под ОС Windows, Linux, MacOS, но мы проверили их работоспособность под управлением OpenSuse Linux 10.2 и 10.3. Не исключено, что при работе под другими ОС возникнут незначительные изменения.

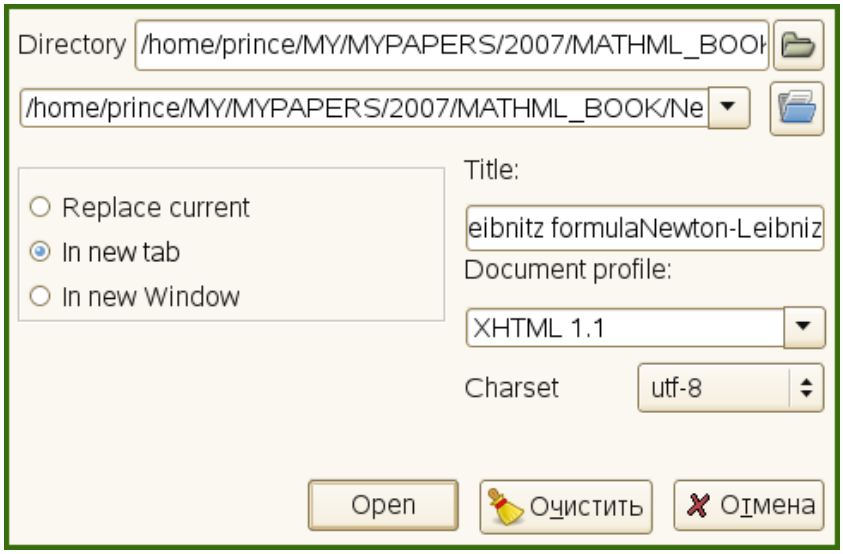
6.1. Редакторы MathML

Amaya (amaya-9.54 версия Feb 14 2007, Linux). Amaya (http://www.w3.org/**Amaya/**) – это веб-редактор, то есть программа, предназначенная для создания и редактирования документов непосредственно в сети. В этой программе предусмотрены возможности просмотра документов и их локального и удаленного редактирования. Работа над этой программой была начата консорциумом W3C в 1996 году с целью продемонстрировать, что инструменты создания веб-документов, использующие различные веб-технологии, можно организовать в одном окружении веб-клиента. Вначале Amaya был редактором HTML + CSS, но потом была добавлена поддержка XML и различных форматов, основанных на XML, в частности, MathML. Мы не будем описывать все возможности данного программного продукта и остановимся лишь на том, как применить его к созданию и редактированию текстов в формате MathML. На странице **http://www.w3.org/Amaya/User/BinDist.html** можно найти дистрибутив Amaya для операционных систем Windows NT/2000/XP/Vista, для основных дистрибутивов Linux и MacOS. Также доступны исходники программы. Установка программы описана на той же странице.

Создание MathML-документа

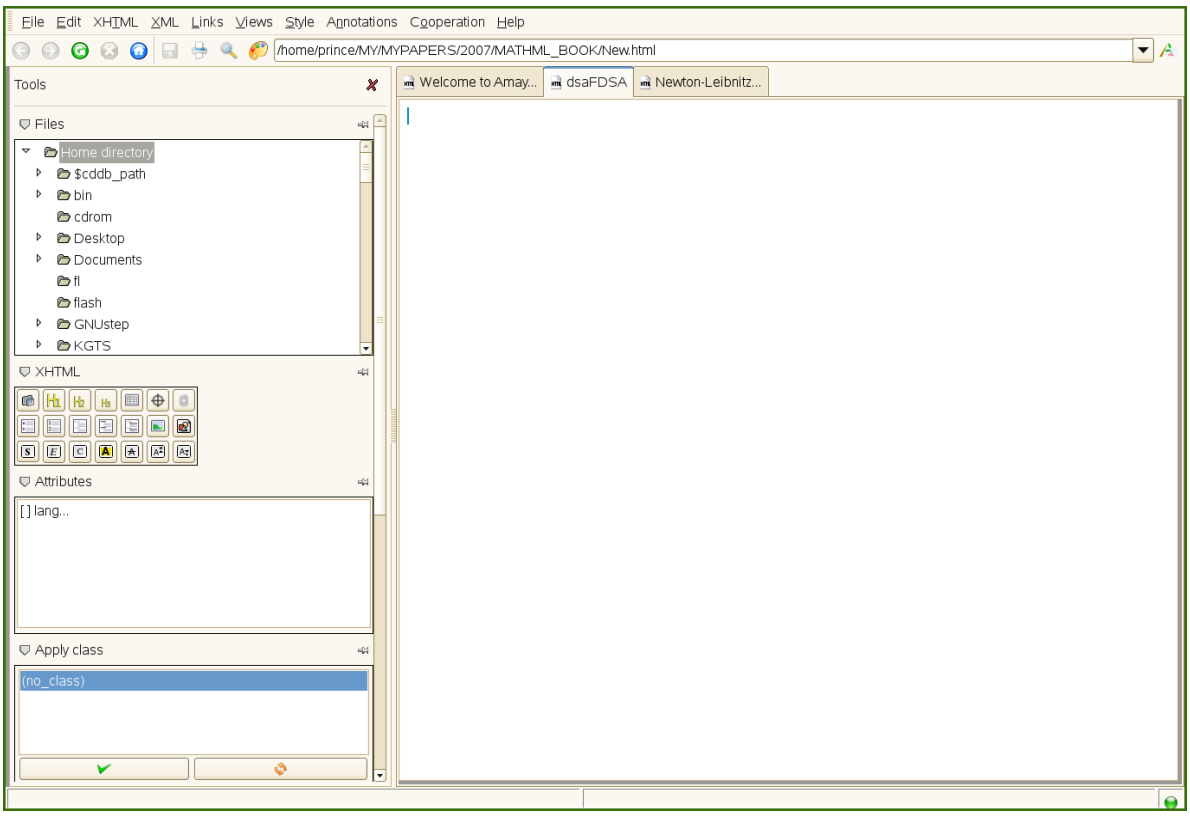


После запуска программы в меню File надо выбрать пункт New и далее «XHTML document». В открывшемся диалоге надо выбрать кодировку utf-8 (мы будем писать по-русски), тип документа XHTML (один из вариантов) и ввести заглавие страницы.

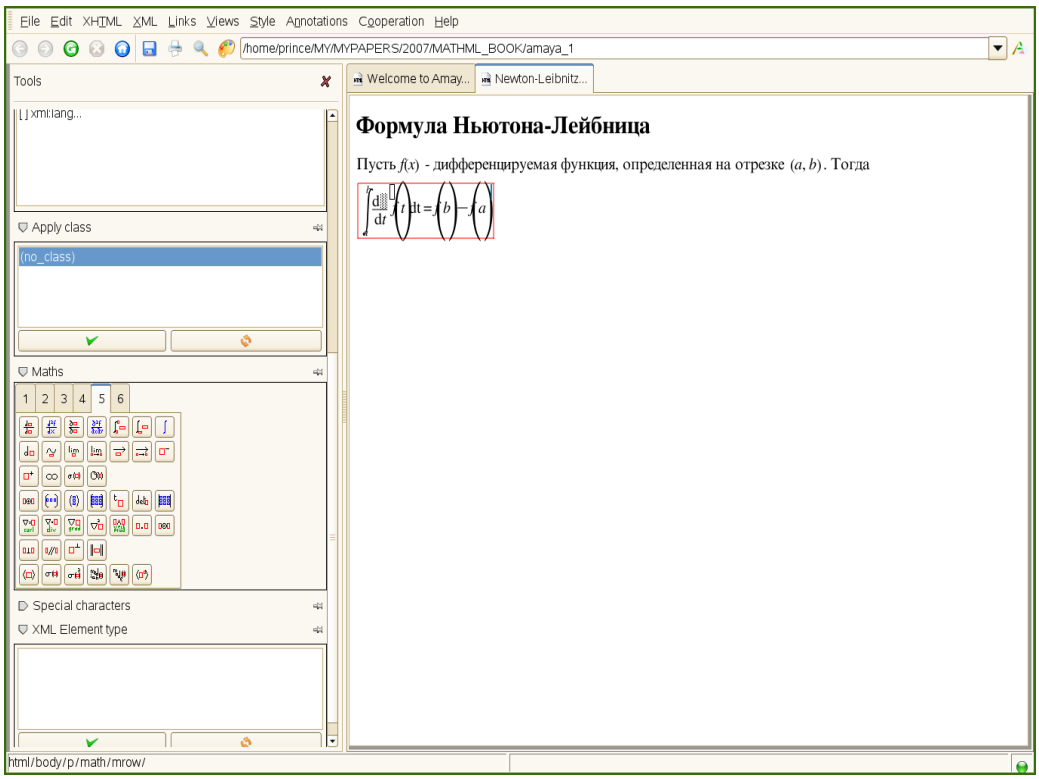


В результате будет создан html-файл. Для работы с MathML-документами лучше сразу открыть панели инструментов (меню Views" Show tools или F8). *Совет: растяните панель инструментов на треть экрана, иначе часть кнопок окажется вне зоны видимости.*

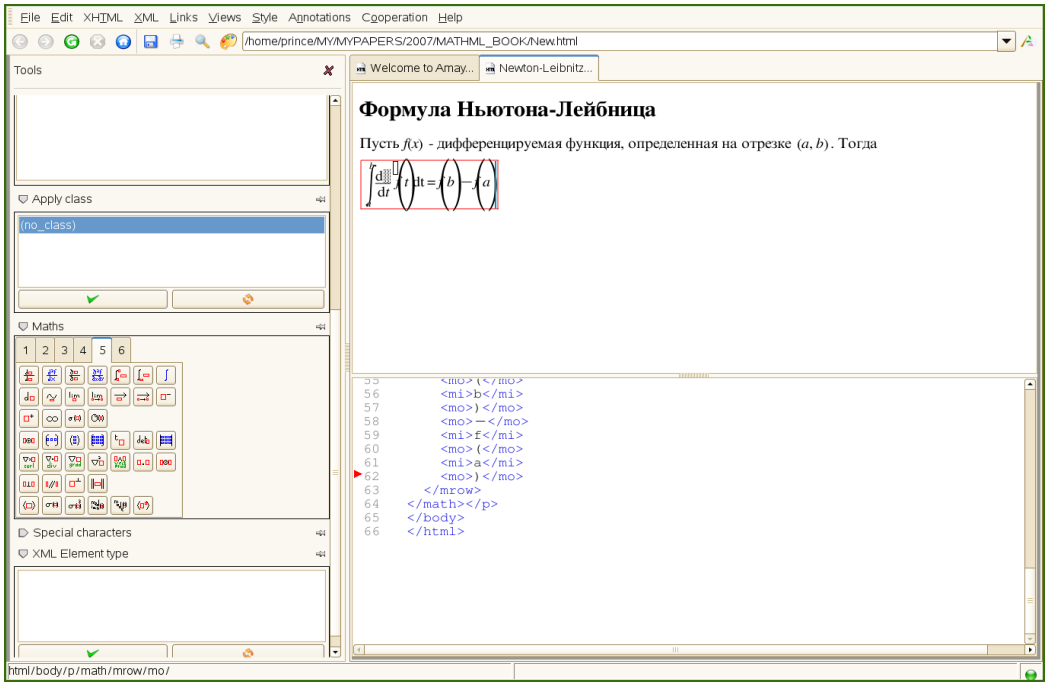
Окно редактора будет выглядеть приблизительно так:



Теперь наберем текст «Формула Ньютона – Лейбница», выделим его и нажмем Ctrl+h Ctrl+1 или выберем соответствующий пункт меню XHTML д Heading У h1. Далее наберем «Пусть», и теперь надо вставить формулу. Для этого нажмем на кнопку Maths на панели инструментов, открытой слева, и далее на кнопку «квадратный корень», расположенную в левом верхнем углу соответствующей панели (или выбрать пункт меню XMLMa New formula, или нажать Ctrl+m Ctrl+m). На экране возникнет затененный прямоугольник красного цвета, который показывает границы формулы. В этом прямоугольнике наберем f(x), а чтобы окончить набор формулы, достаточно два раза нажать правую стрелку. Шрифт формулы на экране автоматически изменится на курсив. Теперь продолжим набирать текст и, перейдя на новую строчку, вновь откроем формулу. В панели Maths перейдем на закладку 5, где найдем все необходимое для набора этой формулы. Набор формулы осуществим подобно любому WYSIWYG-редактору, например, Microsoft Equation. Отметим, что для набора простых математических формул можно пользоваться и меню XMLB Basic Elements, XML O Constructions, XMLw Matrices. В формуле получатся непомерно большие скобки, но это есть недостаток рендеринга браузера Amaya. В остальных браузерах (см. ниже) формула будет выглядеть нормально.

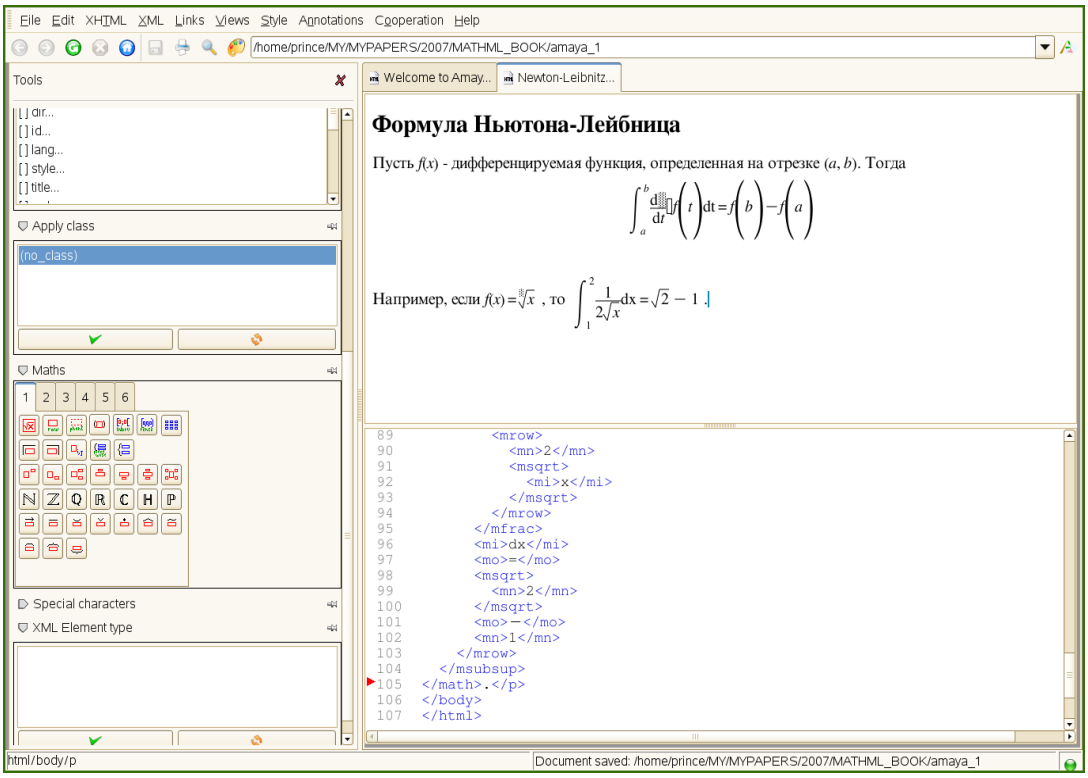


Теперь нам нужно отцентрировать формулу, но на панели инструментов нет для этого средств. Тогда мы откроем исходный текст Views S Show source (Ctrl+u Ctrl+o), и окно браузера будет выглядеть так:



Перейдем в окно с исходным кодом и добавим к соответствующему тэгу <p> атрибут центрирования: получится <p center=" align">. Щелкнув мышкой на окне набора, убедимся, что формула центрирована. Далее продолжим набор, причем здесь удобно пользоваться горячими клавишами, вызывающими соответствующие пункты меню для базовых математических элементов, например, «квадратный корень» – Ctrl+m Ctrl+r. Требуется некоторое время, чтобы при-

выкнуть к набору в этом редакторе, однако в основном все получается достаточно легко. Отметим, что исходный текст в нижнем окне изменится только после сохранения. Получим следующий вид:



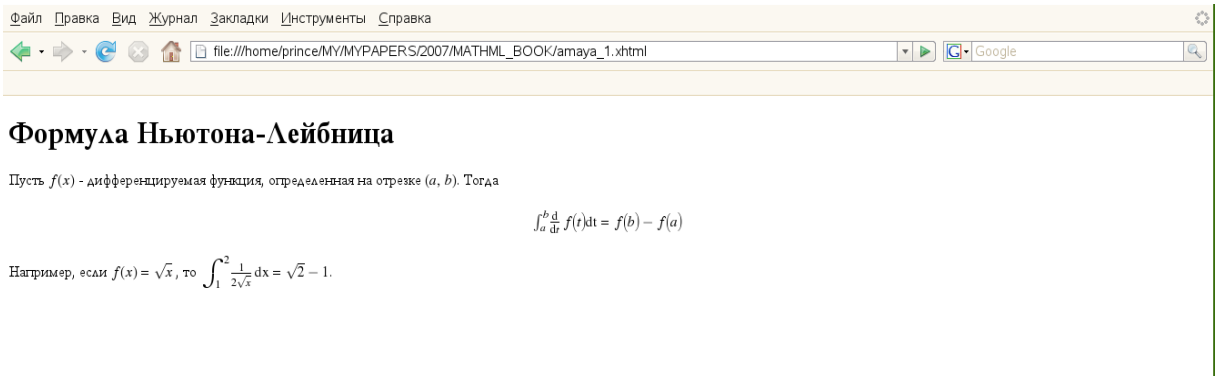
При этом получится следующий MathML-документ:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="pmathml.xsl"?>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
  <title>Newton-Leibnitz formula</title>
  <meta name="generator" content="Amaya 9.54, see http://www.w3.org/Amaya/" />
</head>

<body>
<h1>Формула Ньютона – Лейбница</h1>

<p>Пусть <math xmlns="http://www.w3.org/1998/Math/MathML">
  <mi>f</mi>
  <mo>(</mo>
  <mi>x</mi>
  <mo>)</mo>
</math> – дифференцируемая функция,
определенная на отрезке <math
xmlns="http://www.w3.org/1998/Math/MathML">
  <mo>(</mo>
```

```
  <mi>a</mi>
  <mo>,</mo>
  <mi>b</mi>
  <mo>)</mo>
</math>. Тогда</p>

<p align="center"><math xmlns="http://www.w3.org/1998/Math/MathML">
  <mrow>
    <msubsup>
      <mo>∫</mo>
      <mi>a</mi>
      <mi>b</mi>
    </msubsup>
    <mfrac>
      <mrow>
        <mo>d</mo>
        <mrow>
          <mi></mi>
        </mrow>
      </mrow>
      <mrow>
        <mo>d</mo>
        <mi>t</mi>
      </mrow>
    </mfrac>
    <mi>f</mi>
    <mo>(</mo>
    <mi>t</mi>
    <mo>)</mo>
    <mi>dt</mi>
    <mo>=</mo>
    <mi>f</mi>
    <mo>(</mo>
    <mi>b</mi>
    <mo>)</mo>
    <mo>−</mo>
    <mi>f</mi>
    <mo>(</mo>
    <mi>a</mi>
    <mo>)</mo>
  </mrow>
</math></p>
```

```
<p></p>
<p>Например, если <math
xmlns="http://www.w3.org/1998/Math/MathML">
 <mi>f</mi>
 <mo>(</mo>
 <mi>x</mi>
 <mo>)</mo>
 <mo>=</mo>
 <mroot>
  <mi>x</mi>
  <mi></mi>
 </mroot>
</math> , то <math xmlns="http://www.w3.org/1998/Math/MathML">
    <mrow>
    <msubsup>
     <mo>∫</mo>
     <mn>1</mn>
     <mn>2</mn>
    </msubsup>
    <mfrac>
     <mn>1</mn>
     <mrow>
      <mn>2</mn>
      <msqrt>
       <mi>x</mi>
      </msqrt>
     </mrow>
    </mfrac>
    <mi>dx</mi>
    <mo>=</mo>
    <msqrt>
     <mn>2</mn>
    </msqrt>
    <mo>−</mo>
    <mn>1</mn>
   </mrow>
 </math>.</p>
</body>
</html>
```

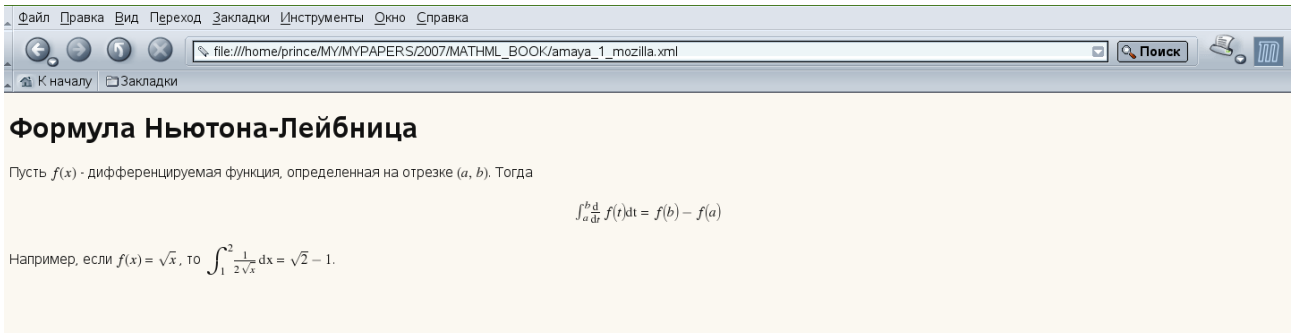
Назовем этот файл amaya_1 и сохраним его с расширением .xml или .xhtml, так как с расширением .html он не будет правильно интерпретироваться браузером. Браузер Firefox 2.0 отображает amaya_1.xhtml следующим образом:



Так полученный файл адаптирован для просмотра с помощью IE (см. раздел «Особенности отображения в различных браузерах»), браузер Mozilla 1.7.11 не отображает этот файл и выдает ошибку «ошибка загрузки стилей». Нужно поправить исходный текст, убрав строку

<?xml-stylesheet type="text/xsl" href="pmathml.xsl"?>,

и Mozilla правильно отобразит данный текст:

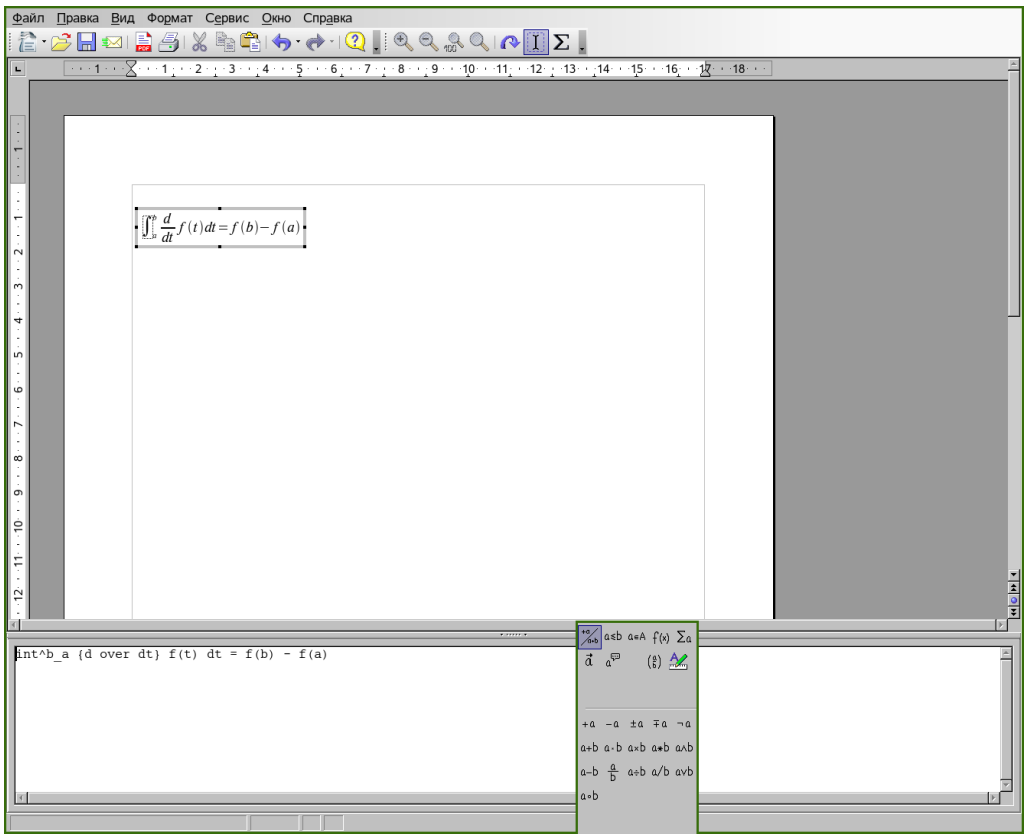


Интересно, что Firefox 2.0 правильно отобразит оба MathML-документа.

В целом складывается ощущение, что веб-редактор Amaya хорош для набора небольших текстов, содержащих несложные математические формулы. Также его можно применять для правки MathML-текстов, созданных другими программными средствами.

6.2. Open Office 2.0.4

Тектовый редактор Open Office Writer, входящий в пакет Open Office, позволяет вставлять в текст математические формулы, подготовленные с помощью редактора формул, входящего в тот же пакет. Для этого надо вставить объект «Формула» (пункт меню: Вставка B Объект Формула). Набор в редакторе формул несколько отличается от набора в Microsoft Equation, открываются панель, позволяющая выбрать различные элементы формул, и текстовое окно, где необходимо набирать TeX-подобный текст.



При нажатии на кнопки в текстовом окне появляются шаблоны формул, например, при нажатии кнопки a/b появляется шаблон «<?> over <?>», в котором вместо <?> надо записать числитель и знаменатель дроби. Такой способ достаточно прост для набора, особенно для пользователей, знакомых с системой TeX.

Полученный текст можно экспортировать в формат HTML, однако при этом формулы будут отображаться картинками. Экспорт в формат XHTML тоже ничего не дает. Однако, если навести указатель мыши на формулу и нажать правую кнопку, то появится меню, в котором есть пункт «Сохранить копию как». При выборе этого пункта открывается диалоговое окно, где надо выбрать тип файла MathML. Будет создан файл с расширением mml, содержащий разметку, адаптированную для IE+MathPlayer. Тем не менее, Firefox 2.0 и Mozilla 1.7.11 правильно отображают этот файл без всякого плагина, если этот файл сохранен с расширением xml. Например, формула (в нотации редактора формул OpenOffice)

int^b_a {d over dt} f(t) dt = f(b) - f(a)

дает следующий текст:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE math:math PUBLIC "-//OpenOffice.org//DTD Modified W3C MathML 1.01//EN" "math.dtd">
<math:math xmlns:math="http://www.w3.org/1998/Math/MathML">
```

```
        <math:semantics>
         <math:mrow>
          <math:mrow>
           <math:msubsup>
            <math:mo math:stretchy="false">∫</math:mo>
            <math:mi>a</math:mi>
            <math:mi>b</math:mi>
           </math:msubsup>
           <math:mfrac>
            <math:mi>d</math:mi>
            <math:mi math:fontstyle="italic">dt</math:mi>
           </math:mfrac>
          </math:mrow>
          <math:mi>f</math:mi>
          <math:mrow>
           <math:mo math:stretchy="false">(</math:mo>
           <math:mi>t</math:mi>
           <math:mo math:stretchy="false">)</math:mo>
          </math:mrow>
          <math:mrow>
           <math:mi math:fontstyle="italic">dt</math:mi>
           <math:mo math:stretchy="false">=</math:mo>
           <math:mi>f</math:mi>
          </math:mrow>
          <math:mrow>
           <math:mrow>
            <math:mo math:stretchy="false">(</math:mo>
            <math:mi>b</math:mi>
            <math:mo math:stretchy="false">)</math:mo>
           </math:mrow>
           <math:mo math:stretchy="false">−</math:mo>
           <math:mi>f</math:mi>
          </math:mrow>
          <math:mrow>
           <math:mo math:stretchy="false">(</math:mo>
           <math:mi>a</math:mi>
           <math:mo math:stretchy="false">)</math:mo>
          </math:mrow>
         </math:mrow>
         <math:annotation math:encoding="StarMath 5.0">int^b_a {d over dt} f(t) dt = f(b) -
f(a)  </math:annotation>
        </math:semantics>
       </math:math>
```

Таким образом, с помощью OpenOffice можно конвертировать формулы в MathML, а потом вставлять их в xml- или xhtml-файл, подготовленный с помощью того же редактора Open Office Writer. Однако это трудоемкий процесс, который имеет смысл только при небольшом количестве формул.

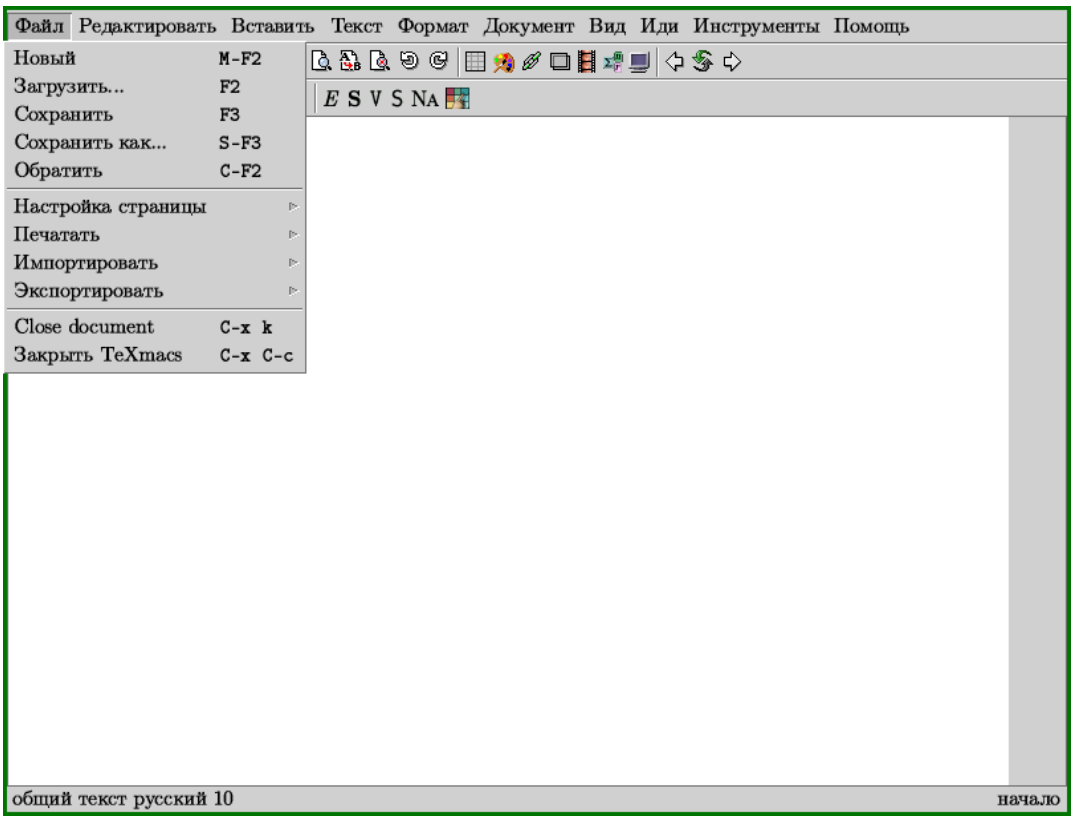
Итак, мы можем подготавливать математические тексты с простыми формулами, например, задания для школьников или студентов, с помощью WYSIWYG-редакторов, и публиковать их в сети.

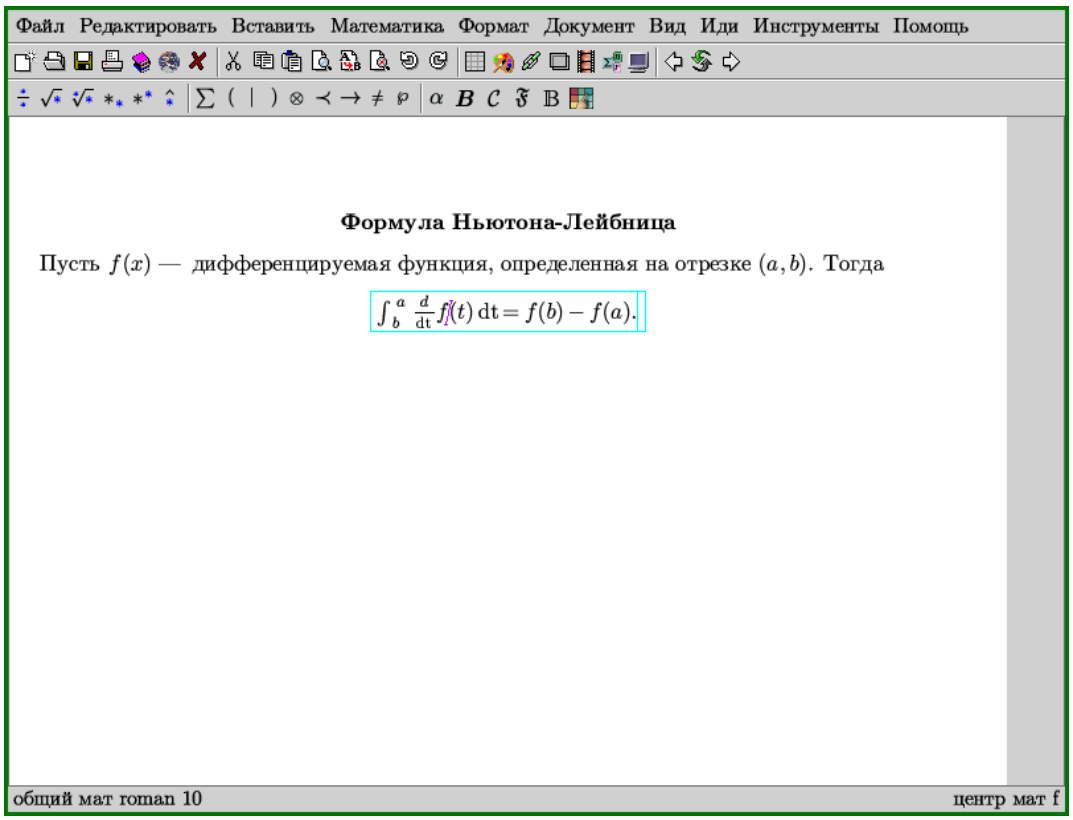
6.3. TeXmacs

GNU TeXmacs (http://www.texmacs.org/tmweb/home/welcome.en.html) – это свободно распространяемая программная среда, специально предназначенная для обработки научных текстов. Она включает в себя WYSIWYG-редактор с поддержкой математических формул и редактор изображений. TeXmacs также может быть использован как front-end (интерфейс) для пакетов программ компьютерной алгебры, численных методов, статистики и т. п. Как и все серьезные научные программные продукты, Texmacs является кроссплатформенным, то есть может работать под управлением Windows, Linux и MacOS. Мы описываем Texmacs (1.0.6.9) под управлением SUSE 10.3 (пакет включен в дистрибутив).

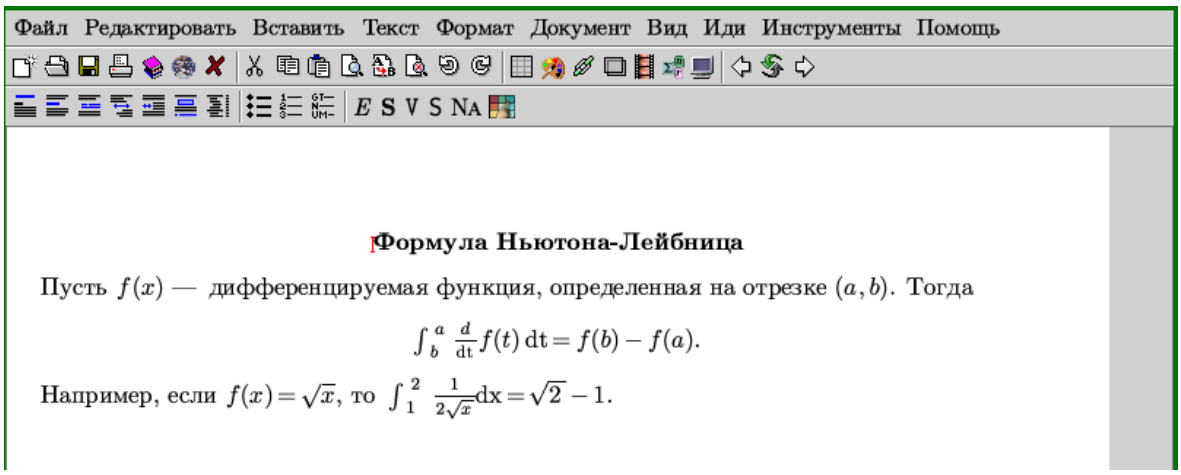
Texmacs имеет собственный формат файлов, которые экспортируются в различный форматы, включая MathML. Набор математических текстов в программной среде TeXmacs поистине впечатляет и, наверное, является идеальным для любителей принципа WYSIWYG. Здесь мы не будем подробно описывать работу в TeXmacs, а лишь покажем, как с его помощью решить нашу задачу: набрать простой текст в MathML.

Чтобы получить искомый текст в MathML, запустив Texmacs, выберем пункт меню Файл У Новый. Так как мы собираемся набирать на русском языке, выберем Редактировать □ Предпочтения У Язык □ Русский. Далее начнем набор, причем операции форматирования осуществляются интуитивно понятным образом с помощью кнопок на панели (советуем читателю познакомиться с этими кнопками повнимательнее – там довольно много интересного). Переход к набору формул определяется вводом знака $, при этом сам знак не печатается, а на экран сразу выводится формула в «откомпилированном» виде. Переход к набору обычного текста происходит после повторного ввода $. Когда мы находимся в режиме ввода математического текста, верхняя панель изменяется, и на ней появляются кнопки для различных математических символов.





Итак, мы получили следующий вид:



Теперь нужно экспортировать его в MathML. По умолчанию экспорт в MathML не установлен, поэтому надо выбрать пункт меню

Редактировать % Предпочтения □ Преобразователи □ TeXmacs B Html w Use MathML.

После этого можно осуществить экспорт, выбрав

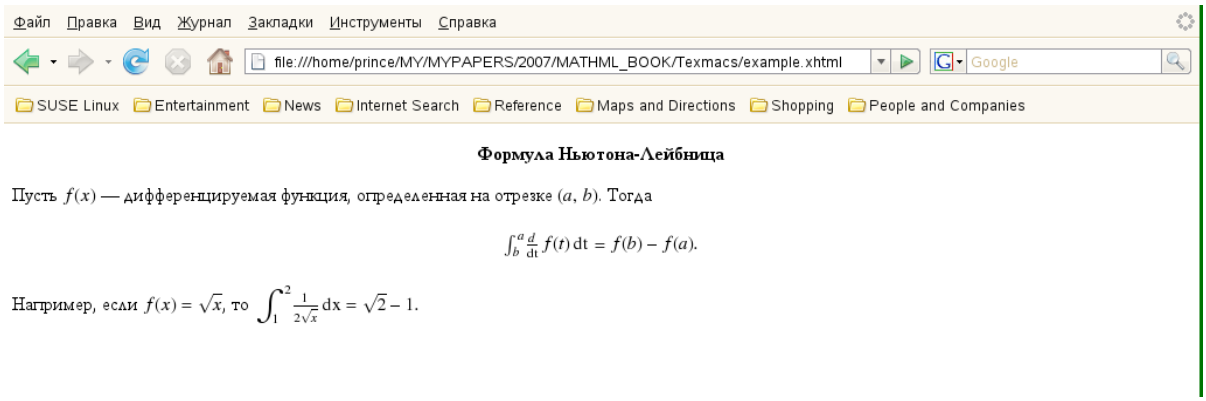
Файл д Экспортировать S Html

и далее взяв расширение xhtml. Обратите внимание, что диалог с пользователем идет внизу экрана – в статусной строке – пользователям Emacs это привычно. Приведем здесь полученный example.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1 plus MathML 2.0//EN"
"http://www.w3.org/TR/MathML2/dtd/xhtml-math11-f.dtd">
<html xmlns:x="http://www.texmacs.org/2002/extensions"
xmlns:m="http://www.w3.org/1998/Math/MathML"
xmlns="http://www.w3.org/1999/xhtml">
 <head>
  <title>No title</title>
  <meta name="generator" content="TeXmacs 1.0.6.9"></meta>
  <style type="text/css">
   body { text-align: justify } h5 { display: inline; padding-right: 1em }
   h6 { display: inline; padding-right: 1em } table { border-collapse:
   collapse } td { padding: 0.2em; vertical-align: baseline } .subsup {
   display: inline; vertical-align: -0.2em } .subsup td { padding: 0px;
   text-align: left} .fraction { display: inline; vertical-align: -0.8em }
   .fraction td { padding: 0px; text-align: center } .wide { position:
   relative; margin-left: -0.4em } .accent { position: relative;
   margin-left: -0.4em; top: -0.1em } .title-block { width: 100%;
   text-align: center } .title-block p { margin: 0px } .compact-block p {
   margin-top: 0px; margin-bottom: 0px } .left-tab { text-align: left }
   .center-tab { text-align: center } .right-tab { float: right; position:
```

```
          relative; top: -1em } math { font-family: cmr, times, verdana }
        </style>
      </head>
      <body>
        <center>
          <p>
            <strong>&#x0424;&#x043E;&#x0440;&#x043C;&#x0443;&#x043B;&#x0430;
            &#x041D;&#x044C;&#x044E;&#x0442;&#x043E;&#x043D;&#x0430;-
&#x041B;&#x0435;&#x0439;&#x0431;&#x043D;&#x0438;&#x0446;&#x0430;</strong>
          </p>
        </center>
        <p>
          &#x041F;&#x0443;&#x0441;&#x0442;&#x044C; <math
xmlns="http://www.w3.org/1998/Math/MathML"><mrow><mi>f</mi><mrow><mo>(</mo
><mi>x</mi><mo>)</mo></mrow></mrow></math>
          <class style="font-family: Times New Roman">&mdash;</class> <math
xmlns="http://www.w3.org/1998/Math/MathML"><mo>&ApplyFunction;</mo></math>&
#x0434;&#x0438;&#x0444;&#x0444;&#x0435;&#x0440;&#x0435;&#x043D;&#x0446;&#x0
438;&#x0440;&#x0443;&#x0435;&#x043C;&#x0430;&#x044F;
          &#x0444;&#x0443;&#x043D;&#x043A;&#x0446;&#x0438;&#x044F;,
          &#x043E;&#x043F;&#x0440;&#x0435;&#x0434;&#x0435;&#x043B;&#x0435;&#
x043D;&#x043D;&#x0430;&#x044F;
          &#x043D;&#x0430;
          &#x043E;&#x0442;&#x0440;&#x0435;&#x0437;&#x043A;&#x0435; <math xm-
lns="http://www.w3.org/1998/Math/MathML"><mrow><mo>(</mo><mi>a</mi><mo>,</m
o><mi>b</mi><mo>)</mo></mrow></math>.
          &#x0422;&#x043E;&#x0433;&#x0434;&#x0430;
        </p>
        <center>
          <p>
            <math
xmlns="http://www.w3.org/1998/Math/MathML"><mrow></mrow></math><math
xmlns="http://www.w3.org/1998/Math/MathML"><mrow><msubsup><mo>&Integral;</m
o><mi>b</mi><mrow><msub><mrow></mrow><mrow></mrow></msub><mi>a</mi></m
row></msubsup><mfrac><mi>d</mi><mi>dt</mi></mfrac><mi>f</mi><mrow><mo>(</m
o><mi>t</mi><mo>)</mo></mrow><mo>&ApplyFunction;</mo><mi>dt</mi><mo>&Ap-
plyFunction;</mo><mo>=</mo><mo>&ApplyFunction;</mo><mi>f</mi><mrow><mo>(</
mo><mi>b</mi><mo>)</mo></mrow><mo>&ApplyFunction;</mo><mo>-
</mo><mo>&ApplyFunction;</mo><mi>f</mi><mrow><mo>(</mo><mi>a</mi><mo>)</
mo></mrow><mi>.</mi></mrow></math>

          </p>
        </center>
```

```
        <p>
          &#x041D;&#x0430;&#x043F;&#x0440;&#x0438;&#x043C;&#x0435;&#x0440;,
          &#x0435;&#x0441;&#x043B;&#x0438; <math
xmlns="http://www.w3.org/1998/Math/MathML"><mrow><mi>f</mi><mrow><mo>(</mo
><mi>x</mi><mo>)</mo></mrow><mo>=</mo><msqrt><mi>x</mi></msqrt></mrow></
math>,
          &#x0442;&#x043E; <math
xmlns="http://www.w3.org/1998/Math/MathML"><mrow><msubsup><mo>&Integral;</m
o><mn>1</mn><mn>2</mn></msubsup><mo>&ApplyFunction;</mo><mfrac><mn>1</m
n><mrow><mn>2</mn><msqrt><mi>x</mi></msqrt></mrow></mfrac><mi>dx</mi><mo
>&ApplyFunction;</mo><mo>=</mo><mo>&ApplyFunction;</mo><mroot><mrow><mn
>2</mn><mo>&ApplyFunction;</mo></mrow><mrow></mrow></mroot><mo>-
</mo><mn>1</mn><mi>.</mi></mrow></math>
        </p>
      </body>
    </html>
```



Обратите внимание, что интеграл в нижней строке получился не совсем удачным.

Непосредственное редактирование текстовых блоков на русском языке в MathML файле затруднительно, поскольку кириллические символы представляются специальными символами HTML. Например, «Формула Ньютона – Лейбница» записывается как

```
&#x0424;&#x043E;&#x0440;&#x043C;&#x0443;&#x043B;&#x0430;
&#x041D;&#x044C;&#x044E;&#x0442;&#x043E;&#x043D;&#x0430;-
&#x041B;&#x0435;&#x0439;&#x0431;&#x043D;&#x0438;&#x0446;&#x0430;
```

6.4. ITEX2MML – простой конвертор

Itex2MML – консольная утилита, которая преобразует текст, содержащий формулы в нотации WebTeX – TeX-подобного языка разметки, предназначенного для подготовки текстов для публикации в сети; см.

http://stuff.mit.edu/afs/athena/software/webeq/currenthome/docs/webtex/webtex.html

в текст, содержащий формулы в MathML нотации. При этом остальной текст не изменяется.

WebTeX был разработан для Java-апплета WebEQ, который позволяет достаточно просто включать авторам математические выражения в веб-страницы. Так как этот апплет является коммерческим продуктом, мы не останавливаемся на нем, читатель может узнать подробности по ссылке

http://stuff.mit.edu/afs/athena/software/webeq/currenthome/docs/welcome.html.

Мы используем версию программы itex2MML 1.3.3 от 21.11.2007. Домашняя страница находится по адресу

http://golem.ph.utexas.edu/~distler/blog/itex2MML.html,

а список команд для IteX, которые во многом совпадают с командами AMSLaTeX, можно найти на странице

http://golem.ph.utexas.edu/~distler/blog/itex2MMLcommands.html.

Предположив, что эта программа установлена, мы можем решить задачу подготовки к публикации выбранного текста следующим образом.

В любом текстовом редакторе наберем следующий текст:

```
<?xml version="1.0"?>
<!DOCTYPE html SYSTEM "mathml.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:math="http://www.w3.org/1998/Math/MathML">
<body>
<h1> Формула Ньютона – Лейбница </h1>
Пусть $f(x)$~--- дифференцируемая функция, определенная на отрезке
$(a,b)$. Тогда
$$
\int^b_a \frac{d}{dt}f(t)\,dt = f(b)-f(a).
$$
Например, если $f(x)=\sqrt{x}$, то
$\int^2_1 \frac{1}{2\sqrt{x}}dx = \sqrt{2}-1$.
</body>
</html>
```

Заметим, что мы вручную указали заголовок xml-документа и добавили стандартную ссылку на таблицу объявлений mathml.dtd, а затем ввели html-раз-

метку. Сам конвертор будет использоваться только для перевода формул в нотации TeX в MathML, то есть itex2MML обрабатывает текст, окруженный знаками «$». При этом конвертор не делает различий между включенными в текст формулами ($...$) и выключенными формулами ($$..$$). Также он не ведет автоматическую нумерацию формул. Поэтому itex2MML лучше использовать для простых документов.

В консоли зададим команду

./itex2MML < itex.tex > itex.xml

(напомним, что мы работаем в SUSE Linux) и получим файл itex.xml

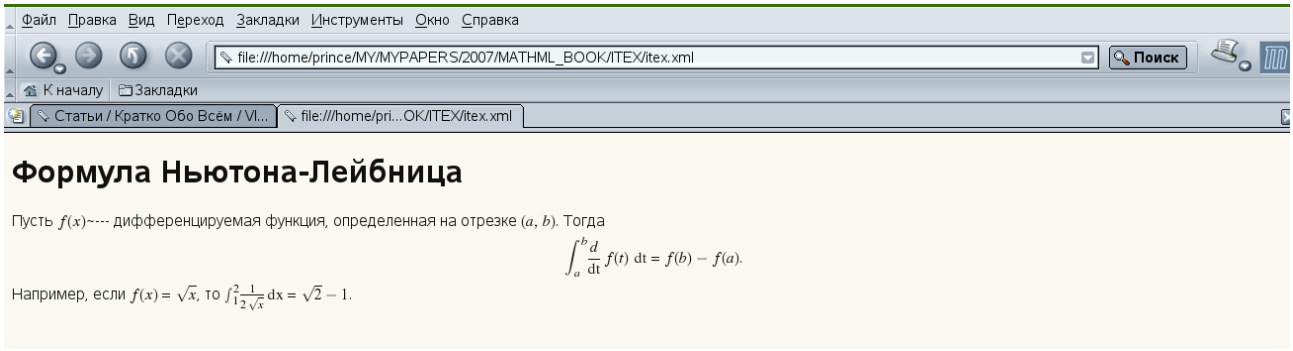
```
<?xml version="1.0"?>
<!DOCTYPE html SYSTEM "mathml.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:math="http://www.w3.org/1998/Math/MathML">
<body>
<h1> Формула Ньютона – Лейбница </h1>
Пусть <math xmlns='http://www.w3.org/1998/Math/MathML'
display='inline'><mi>f</mi><mo stretchy="false">(</mo><mi>x</mi><mo
stretchy="false">)</mo></math>~--- дифференцируемая функция, определенная на от-
резке
<math xmlns='http://www.w3.org/1998/Math/MathML' display='inline'><mo
stretchy="false">(</mo>
<mi>a</mi><mo>,</mo><mi>b</mi><mo stretchy="false">)</mo></math>.
Тогда
<math xmlns='http://www.w3.org/1998/Math/MathML'
display='block'><msubsup><mo>&Integral;</mo>
<mi>a</mi>
<mi>b</mi></msubsup><mfrac><mi>d</mi><mi>dt</mi></mfrac><mi>f</mi><mo
stretchy="false">(</mo>
<mi>t</mi><mo stretchy="false">)</mo><mspace
width="thinmathspace"></mspace><mi>dt</mi><mo>=</mo><mi>f
</mi><mo stretchy="false">(</mo><mi>b</mi><mo
stretchy="false">)</mo><mo>&minus;</mo><mi>f</mi>
<mo stretchy="false">(</mo><mi>a</mi><mo
stretchy="false">)</mo><mo>.</mo></math>
Например, если <math xmlns='http://www.w3.org/1998/Math/MathML' display='in-
line'><mi>f</mi><mo stretchy="false">(</mo><mi>x</mi>
<mo stretchy="false">)</mo><mo>=</mo><msqrt><mi>x</mi></msqrt></math>,
то
<math xmlns='http://www.w3.org/1998/Math/MathML'
display='inline'><msubsup><mo>&Integral;</mo> <mn>1
</mn> <mn>2 </mn></msubsup><mfrac><mn>1 </mn><mrow><mn>2
</mn><msqrt><mi>x</mi></msqrt></mrow></mfrac><mi>dx</mi>
```

```
      <mo>=</mo><msqrt><mn>2 </mn></msqrt><mo>&minus;</mo><mn>1
</mn></math>.
      </body>
      </html>
```

Обратим внимание, что в MathML-файле используются сущности ∫ и −, поэтому необходимо объявление

<!DOCTYPE html SYSTEM "mathml.dtd">,

иначе браузер выдаст ошибку (что эти сущности не определены).



6.5. Конвертор tex4ht

Пакет tex4ht (http://www.cse.ohio-state.edu/~gurari/TeX4ht/), созданный профессором университета Огайо Eitan M. Gurari, является полнофункциональным конвертором из TeX в различные веб-форматы. Мы рассмотрим только конвертацию TeX в MathML. Названный пакет включен во все основные дистрибутивы Linux. Также инсталлятор tex4ht для Windows, Linux и MacOS можно найти в сети, в частности, на странице

http://www.cse.ohio-state.edu/~gurari/TeX4ht/mn22.html.

Пакет tex4ht можно скомпилировать и из исходных текстов, ссылка на которые находится на той же странице. Например, в дистрибутив SUSE 10.* пакет tex4ht не входит, но подробная инструкция по компиляции и установке находится по адресу http://www.exstrom.com/journal/comp/tex4ht.html.

Опишем работу с пакетом tex4ht для системы MikTeX (работа с MathZilla аналогична). Для упрощения обозначений будем считать, что все программы установлены в стандартных каталогах на диске C:.

Установка пакета TeX4ht предусматривает следующие действия:

- скачать файл **tex4ht.zip** с указанного веб-адреса;
- создать каталог **Tex4ht**;
- распаковать файл **tex4ht.zip** в каталог **Tex4ht**;
- все файлы из каталога **tex4ht\texmf\tex\generic\tex4ht** переместить в каталог **C:\tex4ht\bin\win32**;
- в файле **C:\tex4ht\texmf\tex4ht\base\win32\tex4ht.env** строку **tc:\path\tfm!**

заменить на строку **tc:\texmf\fonts\tfm!;**

- проверьте: если каталог **C:\tex4ht\texmf\tex4ht\htfonts** содержит подкаталог **htfonts**, то содержимое этого подкаталога надо переместить в **C:\tex4ht\texmf\tex4ht\.**

После установки пакета конвертация TeX-файла осуществляется одной командой. Например, наш файл example.tex конвертируется в example.xml для просмотра браузером Mozilla с помощью команды **mzlatex example.tex.** В результате создается файл example.xml следующего вида.

```
<?xml version="1.0" encoding="iso-8859-5" ?>
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1 plus MathML 2.0//RU"
    "http://www.w3.org/Math/DTD/mathml2/xhtml-math11-f.dtd" >
    <?xml-stylesheet type="text/css" href="example.css"?>
    <html lang="ru"
    xmlns="http://www.w3.org/1999/xhtml"
    ><head><title></title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-5" />
    <meta name="generator" content="TeX4ht (http://www.cse.ohio-
state.edu/~gurari/TeX4ht/)" />
    <meta name="originator" content="TeX4ht (http://www.cse.ohio-
state.edu/~gurari/TeX4ht/)" />
    <!-- xhtml,mozilla -->
    <meta name="src" content="example.tex" />
    <meta name="date" content="2007-11-29 15:42:00" />
    <link rel="stylesheet" type="text/css" href="example.css" />
    </head><body
    >
     <h3 class="sectionHead"><span class="titlemark">1. </span> <a
     id="x1-10001"></a>&#x0424;&#x043E;&#x0440;&#x043C;&#x0443;&#x043B;&#x0
430; &#x041D;&#x044C;&#x044E;&#x0442;&#x043E;&#x043D;&#x0430;-
&#x041B;&#x0435;&#x0439;&#x0431;&#x043D;&#x0438;&#x0446;&#x0430;</h3>
    <!--l. 8--><p class="noindent" >&#x041F;&#x0443;&#x0441;&#x0442;&#x044C; <!--
l. 8--><math
     xmlns="http://www.w3.org/1998/Math/MathML" display="inline" ><mi
    >f</mi><mrow ><mo
    class="MathClass-open">(</mo><mrow><mi
    >x</mi></mrow><mo
    class="MathClass-close">)</mo></mrow></math> &#x2014;
    &#x0434;&#x0438;&#x0444;&#x0444;&#x0435;&#x0440;&#x0435;&#x043D;&#x044
6;&#x0438;&#x0440;&#x0443;&#x0435;&#x043C;&#x0430;&#x044F;
&#x0444;&#x0443;&#x043D;&#x043A;&#x0446;&#x0438;&#x044F;,
&#x043E;&#x043F;&#x0440;&#x0435;&#x0434;&#x0435;&#x043B;&#x0435;&#x043D;&
#x043D;&#x0430;&#x044F; &#x043D;&#x0430;
```

```
&#x043E;&#x0442;&#x0440;&#x0435;&#x0437;&#x043A;&#x0435;
      <!--l. 9--><math
       xmlns="http://www.w3.org/1998/Math/MathML" display="inline" ><mrow ><mo
      class="MathClass-open">(</mo><mrow><mi
      >a</mi><mo
      class="MathClass-punc">,</mo><mi
      >b</mi></mrow><mo
      class="MathClass-close">)</mo></mrow></math>.
      &#x0422;&#x043E;&#x0433;&#x0434;&#x0430;
      <!--tex4ht:inline--></p><!--l. 10--><math
       xmlns="http://www.w3.org/1998/Math/MathML" display="block" >
                    <msubsup><mrow
      ><mo
      class="MathClass-op">&#x222B;
       <!--nolimits--></mo><!--nolimits--></mrow><mrow
      ><mi
      >a</mi></mrow><mrow
      ><mi
      >b</mi></mrow></msubsup
      > <mfrac><mrow
      ><mi
      >d</mi></mrow>
      <mrow
      ><mi
      >d</mi><mi
      >t</mi></mrow></mfrac><mi
      >f</mi><mrow ><mo
      class="MathClass-open">(</mo><mrow><mi
      >t</mi></mrow><mo
      class="MathClass-close">)</mo></mrow><mspace width="0em"
class="thinspace"/><mi
      >d</mi><mi
      >t</mi> <mo
      class="MathClass-rel">=</mo> <mi
      >f</mi><mrow ><mo
      class="MathClass-open">(</mo><mrow><mi
      >b</mi></mrow><mo
      class="MathClass-close">)</mo></mrow> <mo
      class="MathClass-bin">&#x2212;</mo> <mi
      >f</mi><mrow ><mo
      class="MathClass-open">(</mo><mrow><mi
      >a</mi></mrow><mo
      class="MathClass-close">)</mo></mrow><mo
```

```
      class="MathClass-punc">.</mo>
      </math>
      <!--l. 12--><p class="nopar" >
&#x041D;&#x0430;&#x043F;&#x0440;&#x0438;&#x043C;&#x0435;&#x0440;,
&#x0435;&#x0441;&#x043B;&#x0438; <!--l. 13--><math
       xmlns="http://www.w3.org/1998/Math/MathML" display="inline" ><mi
      >f</mi><mrow ><mo
      class="MathClass-open">(</mo><mrow><mi
      >x</mi></mrow><mo
      class="MathClass-close">)</mo></mrow> <mo
      class="MathClass-rel">=</mo> <msqrt><mrow><mi
      >x</mi></mrow></msqrt></math>,
      &#x0442;&#x043E; <!--l. 14--><math
       xmlns="http://www.w3.org/1998/Math/MathML" display="inline"
><msubsup><mrow
      ><mo
      class="MathClass-op"> &#x222B;
       <!--nolimits--></mo><!--nolimits--></mrow><mrow
      ><mn>1</mn></mrow><mrow
      ><mn>2</mn></mrow></msubsup
      >  <mfrac><mrow
      ><mn>1</mn></mrow>
      <mrow
      ><mn>2</mn><msqrt><mrow><mi
      >x</mi></mrow></msqrt></mrow></mfrac><mi
      >d</mi><mi
      >x</mi> <mo
      class="MathClass-rel">=</mo> <msqrt><mrow><mn>2</mn></mrow></msqrt>
<mo
      class="MathClass-bin">&#x2212;</mo> <mn>1</mn></math>.
      </p>

      </body>
      </html>
```

Отметим, что в данном файле используется файл таблицы стилей example.css. Также обратим внимание на то, что русские буквы представляются специальными символами HTML, а кодировка поставлена iso-8859-5.

Если теперь дать команду

mzlatex example_1.tex "html, mathplayer"

то будет создан файл example.xht, который будет просматриваться через IE +MathPlayer. Отметим, что он также просматривается браузером Firefox. Конвертор Tex4ht прекрасно конвертирует TeX-документы любого объема и слож-

ности.

Если полученный XML-файл содержит ошибки, то они, как правило, вызваны неправильными конструкциями в исходном TeX-файле. В основном такие ошибки связаны с ошибками в расстановке ограничителей, к которым компилятор TeX относится “толерантно” и исправляет их так, что на выходе они становятся незаметными. Например, фрагмент кода

```
$f(x)=g(x$)
```

не вызывает сообщений компилятора TeX об ошибке, однако приводит к сбою при конвертации – получается следующий MathML-код:

```
<mi>f</mi>
<mrow ><mo class="MathClass-open">(</mo>
<mrow><mi>x</mi></mrow>
<mo class="MathClass-close">)</mo>
</mrow>
<mo class="MathClass-rel">=</mo>
<mi>g</mi>
<mrow ><mo class="MathClass-open">(</mo>
<mrow><mi>x</mi>)
```

котором тэг mrow оказывается не закрыт. Такие же проблемы могут возникать в многострочных выключенных формулах, например,

```
\begin{multline}
f(x) = \int [ x^2 + \dots
\\
z^2 ] du
\end{multline}
```

будет воспринят спокойно компилятором TeX, но на самом деле по правилам TeX этот абзац должен быть набран так:

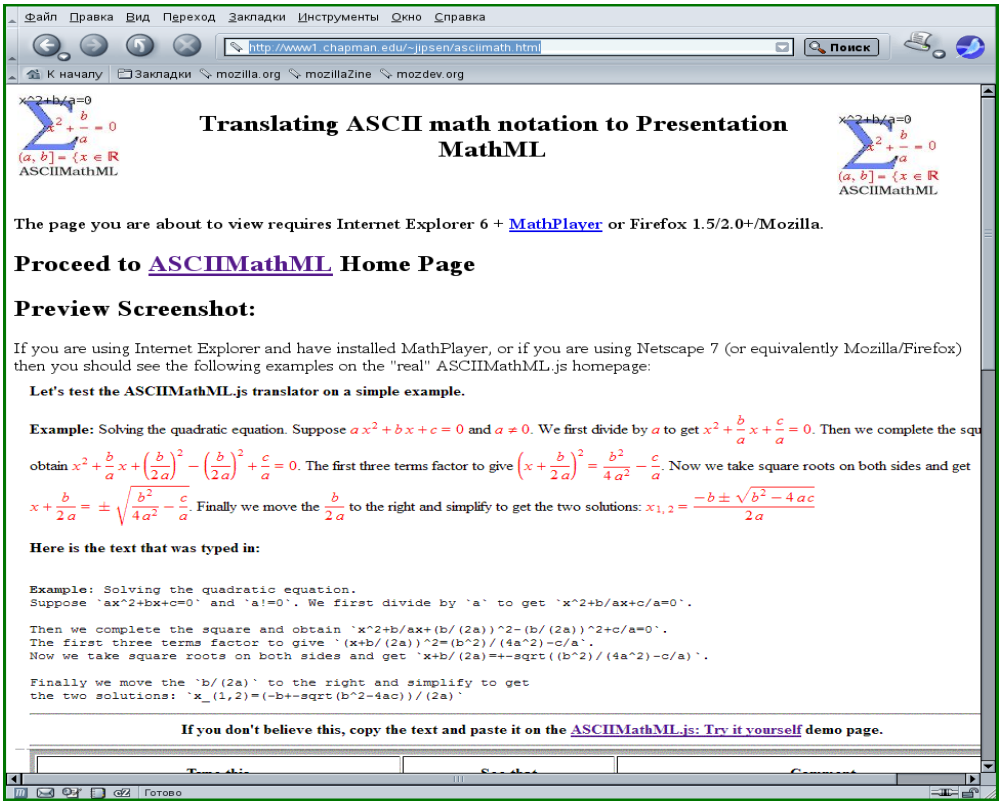
```
\begin{multline}
f(x) = \int \left[ x^2 + \dots \right.
\\
\left. z^2 ] du
\end{multline}
```

Таким образом, любители изобретать свои собственные «приемчики» набора (не понимающие, что такое технологическая культура, в данном случае, культура набора текстов в разметке TeX) могут столкнуться с проблемами. В любом случае можно порекомендовать после набора Tex-файла воспользоваться программами chktex или lacheck для его проверки, даже если вы не собираетесь его конвертировать. Ошибок (в том числе и математических) в тексте станет меньше.

6.6. On-line конверторы в MathML.

В настоящее время в интернете представлен ряд конверторов в формат MathML, которые позволяют произвести конвертацию в режиме on-line. Рассмотрим в качестве примера два из них.

ASCIIMathML (http://www1.chapman.edu/~jipsen/asciimath.html)



Этот ресурс использует Java-скрипт ASCIIMathML.js (ver 2.0; сентябрь 2007; http://www1.chapman.edu/~jipsen/mathml/asciimath.html, автор скрипта – Peter Jipsen), работающий на компьютере пользователя, который загружается при загрузке демонстрационной страницы

http://www1.chapman.edu/~jipsen/mathml/asciimathdemo.html.

Поэтому, в частности, этот ресурс может использоваться локально: достаточно сохранить упомянутую демонстрационную html-страницу и можно производить конвертацию простых формул без подключения к интернету.

Непосредственная работа с этим ресурсом выглядит следующим образом: в окне для ввода набирается текст в Tex-подобной нотации (правила набора объяснены на демонстрационной странице и на странице

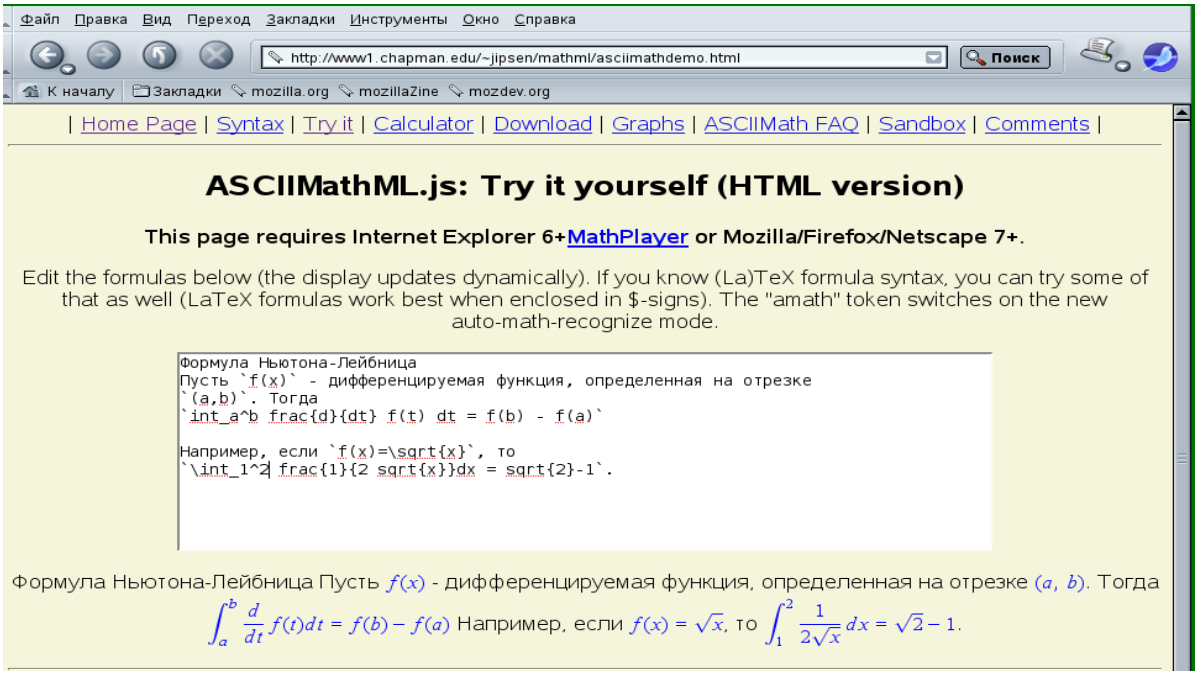
http://www1.chapman.edu/~jipsen/mathml/asciimath.html).

Например, для набора нашего текста необходимо ввести в окно для ввода (обратите внимание, что знак ` используется вместо знака $).

Формула Ньютона - Лейбница
Пусть `f(x)` - дифференцируемая функция, определенная на отрезке `(a,b)`. Тогда

`int_a^b frac{d}{dt} f(t) dt = f(b) - f(a)`
Например, если `f(x)=\sqrt{x}`, то
`\int_1^2 frac{1}{2 sqrt{x}}dx = sqrt{2}-1`.

Результат вполне удовлетворителен (см. рисунок). Следует отметить, что конвертация осуществляется по мере ввода текста, то есть можно исправлять ошибки ввода непосредственно в процессе ввода.



Для того чтобы получить сам MathML-код, например, в браузерах Mozilla и Firefox, нужно в меню, открывающемся при нажатии правой кнопки мыши, выбрать “просмотр исходного кода страницы”. В нашем примере получится следующий код:

<id="outputNode">Формула Ньютона-Лейбница
Пусть <math title="f(x)"><mstyle fontfamily="serif" displaystyle="true" mathcolor="blue"><mrow><mi>f</mi><mrow><mo>(</mo><mi>x</mi><mo>)</mo></mrow></mrow></mstyle></math> - дифференцируемая функция, определенная на отрезке
<math title="(a,b)"><mstyle fontfamily="serif" displaystyle="true" mathcolor="blue"><mrow><mo>(</mo><mi>a</mi><mo>,</mo><mi>b</mi><mo>)</mo></mrow></mstyle></math>. Тогда
<math title="int_a^b frac{d}{dt} f(t) dt = f(b) - f(a)"><mstyle fontfamily="serif" displaystyle="true" mathcolor="blue"><mrow><msubsup><mo>∫</mo><mi>a</mi><mi>b</mi></msubsup></mrow><mfrac><mrow><mi>d</mi></mrow><mrow><mrow><mi>d</mi><mi>t</mi></mrow></mrow></mfrac><mrow><mi>f</mi><mrow><mo>(</mo><mi>t</mi><mo>)</mo></mrow></mrow><mrow><mi>d</mi><mi>t</mi></mrow><mo>=</mo><mrow><mi>f</mi><mrow><mo>(</mo><mi>b</mi><mo>)</mo></mrow></mrow><mo>-

```
</mo><mrow><mi>f</mi><mrow><mo>(</mo><mi>a</mi><mo>)</mo></mrow></mrow>
</mstyle></math></span>
    Например, если <span style="font-size: 1em; font-family: serif;"><math
title="f(x)=\sqrt{x}"><mstyle fontfamily="serif" displaystyle="true"
mathcolor="blue"><mrow><mi>f</mi><mrow><mo>(</mo><mi>x</mi><mo>)</mo></mr
ow></mrow><mo>=</mo><msqrt><mrow><mi>x</mi></mrow></msqrt></mstyle></math
></span>, то
    <span style="font-size: 1em; font-family: serif;"><math title="\int_1^2 frac{1}{2
sqrt{x}}dx = sqrt{2}-1"><mstyle fontfamily="serif" displaystyle="true" math-
color="blue"><mrow><msubsup><mo>∫</mo><mn>1</mn><mn>2</mn></msubsup></mr
ow><mfrac><mrow><mn>1</mn></mrow><mrow><mn>2</mn><msqrt><mrow><mi>x</
mi></mrow></msqrt></mrow></mfrac><mrow><mi>d</mi><mi>x</mi></mrow><mo>=<
/mo><msqrt><mrow><mn>2</mn></mrow></msqrt><mo>-
</mo><mn>1</mn></mstyle></math></span>.</div>
```

Peter Jipsen предназначает свой скрипт для создания динамических xhtml-страниц и для электронных писем и сообщений на форумах, содержащих формулы (см. http://math.chapman.edu/email/). Создадим xhtml-страницу, выводящую наш тест, с использованием скрипта ASCIIMathML. Для этого вначале скачаем файл ASCIIMathML.js. Достаточно сохранить страницу

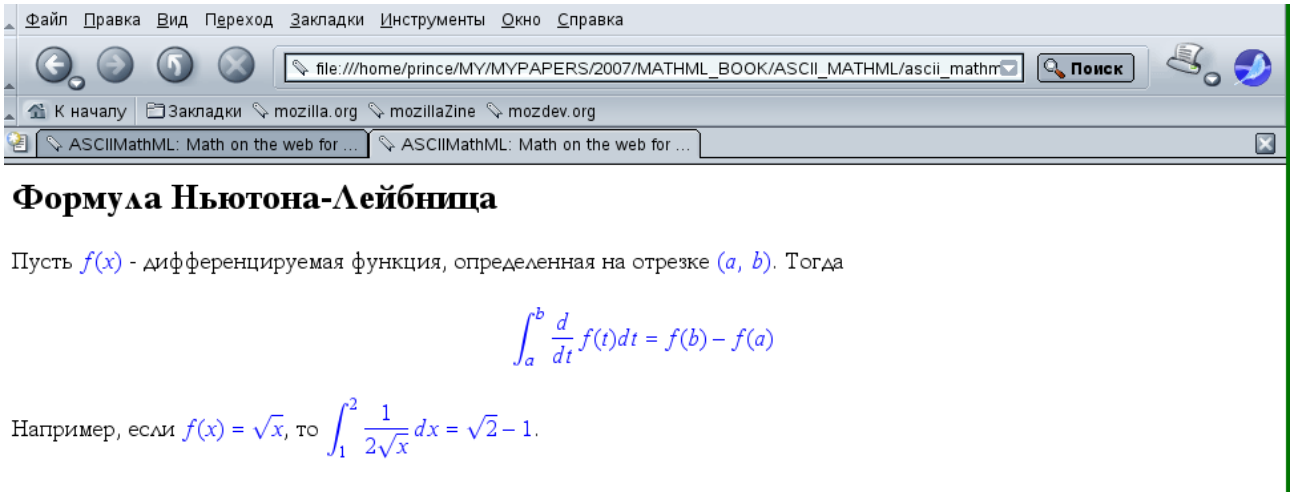
http://www1.chapman.edu/~jipsen/mathml/asciimath.html,

и тогда этот файл скачается автоматически. Теперь создадим html-файл, содержащий следующий текст:

```
<html>
 <head>
  <title>ASCIIMathML: Math on the web for everyone</title>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <script type="text/javascript" src="ASCIIMathML.js"></script>
 <body>
 <h2>Формула Ньютона - Лейбница</h2>
 <p>
  Пусть `f(x)` - дифференцируемая функция, определенная на отрезке
  `(a,b)`. Тогда
 <p>
 <p align=center>
  `int_a^b frac{d}{dt} f(t) dt = f(b) - f(a)`
 </p>
 <p>
  Например, если `f(x)=sqrt{x}`, то
  `int_1^2 frac{1}{2 sqrt{x}}dx = sqrt{2}-1`.
 </p>
 </body>
```

```
</html>
```

и разместим полученный файл в одном каталоге с файлом ASCIIMathML.js. Результат отображения этого файла в браузере представлен на следующем рисунке:



Отметим, что скрипт ASCIIMathML.js может иметь весьма широкое применение. Благодаря тому, что он работает на компьютере пользователя, он может быть использован в разнообразных программных средствах для отображения математических формул в браузере. Отметим, что этот скрипт уже используется в системах Movable Type, WordPress, phpBB и разнообразных вики-энциклопедиях. Также имеется php-порт этого скрипта **ASCIIMathPHP (**подробности см. на странице http://www.oldschool.com.sg/index.php/module/Shared/action/Static/tmpl/ASCIIMathPHP).

7. RDF

7.1. Основные понятия

RDF (Resource Description Framework – среда описания ресурсов сети) [27] – спецификация консорциума W3C для создания моделей, описывающих совокупности данных (ресурсов) интернета с учетом их структуры. Эта спецификация определяет правила описания ресурсов и язык для работы с этими описаниями. Спецификация RDF разработана исключительно для взаимодействия компьютерных программ между собой и не имеет средств отображения информации для людей. RDF предоставляет модель данных, которая проста в обработке приложениями и не зависит от конкретной реализации синтаксиса.

Для описания этой модели введем соответствующие понятия.

URI (Uniform Resource Identifier) – единообразный идентификатор ресурса. Это строка, позволяющая идентифицировать какой-либо ресурс: документ,

изображение, файл, службу, ящик электронной почты и т. д. URI предоставляет простой и расширяемый способ идентификации ресурсов. Расширяемость URI означает, что уже существуют несколько схем идентификации внутри URI, и ещё больше будет создано в будущем. Самые известные примеры URI – это URL и URN. URL – это URI, который помимо идентификации ресурса предоставляет ещё и информацию о местонахождении этого ресурса. URN – это URI, который идентифицирует ресурс в определённом пространстве имён (и, соответственно, в определённом контексте).

Например, URN urn:ISBN 0-395-36341-1 – это URI, который указывает на ресурс (книгу) 0-395-36341-1 в пространстве имён ISBN, но, в отличие от URL, URN не указывает на местонахождение этого ресурса.

В базовом виде URI представляется так:

<схема>:<идентификатор-в-зависимости-от-схемы>.

В этой записи схема – это схема обращения к ресурсу, например, http, ftp, mailto, urn; идентификатор-в-зависимости-от-схемы – непосредственный идентификатор ресурса, вид которого зависит от выбранной схемы обращения к ресурсу

Примеры URI:

http://ru.wikipedia.org/wiki/URI,
ftp://ftp.is.co.za/rfc/rfc1808.txt,
file://C:\UserName.HostName\Projects\Wikipedia_Articles\URI.xml,
ldap://[2001:db8::7]/c=GB?objectClass?one,
mailto:John.Doe@example.com,
sip:911@pbx.mycompany.com,
news:comp.infosystems.www.servers.unix,
data:text/plain;charset=iso-8859-7,%be%fg%be,
tel:+1-816-555-1212,
telnet://192.0.2.16:80/,
urn:oasis:names:specification:docbook:dtd:xml:4.1.2.

Ресурс – это любой объект, имеющий URI. С формальной точки зрения в среде RDF ресурс однозначно характеризуется своим URI.

Литерал – это строка символов.

Триплет – это упорядоченная тройка ресурсов и литералов, элементы которой называются последовательно субъект, предикат и объект (элементы триплета называют иногда «ресурс», «свойство ресурса» и «значение свойства», однако мы не будем использовать эту терминологию). Субъект и предикат – ресурсы, объект – ресурс или литерал.

Триплет предназначен для описания свойств данных в сети. Например, пусть Alexander Vassiljev – автор статьи, выставленной по адресу http://a.b.org. Этот факт может быть описан триплетом

(http://a.b.org, article:author, «Alexander Vassiljev»).

Здесь http://a.b.org и page:author являются URI, а «Alexander Vassiljev» – литералом.

Набор триплетов называют RDF-графом (набором данных RDF, RDF-хранилищем и т. п.). Визуально RDF-граф может быть представлен в виде диаграммы узлов и направленных дуг, в которой каждый триплет представляется в виде связи узел-дуга-узел, однако для машинной обработки больше подходит представление RDF-графа на языке XML.

Приведем пример: опишем номер журнала, состоящий из двух статей J. Smith «Something», A. Parrot «Anything», которые расположены по адресам

www.journal.org/volume1/a.html, www.journal.org/volume1/b.html.

Данный номер может быть описан с помощью RDF-графа, состоящего из триплетов

<www.journal.org/volume1/a.html, journal:author, «J. Smith»>
<www.journal.org/volume1/a.html, journal:title, «Something»>
<www.journal.org/volume1/b.html, journal:author, «A. Parrot»>
<www.journal.org/volume1/b.html, journal:title, «Anything»>

Этот RDF-граф может быть описан с помощью следующего XML-файла:

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:journal="http://www.journal#">
<rdf:Description
rdf:about="www.journal.org/volume1/a.html"/>
<journal:author>J. Smith</journal:author>
<journal:title>Something</journal:title>
</rdf:Description>
<rdf:about="www.journal.org/volume1/b.html"/>
<journal:author>A. Parrot</journal:author>
<journal:title>Anything</journal:title>
</rdf:Description>
</rdf:RDF>
```

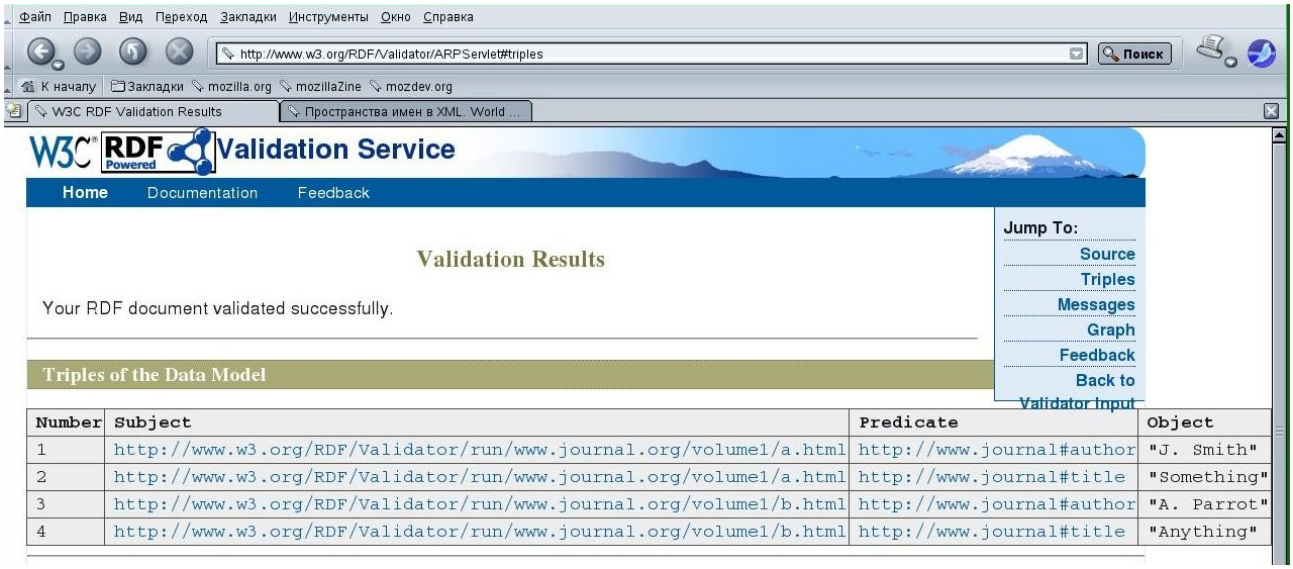
Первая строка этого RDF-документа – XML-декларация. Затем идет корневой элемент RDF-документа <rdf:RDF>. Пространство имен xmlns:rdf определяет, что элементы с префиксом rdf берутся из пространства имен «http://www.w3.org/1999/02/22-rdf-syntax-ns#». Пространство имен xmlns:journal

определяет, что элементы с префиксом journal берутся из пространства имен «http://www.journal#» .

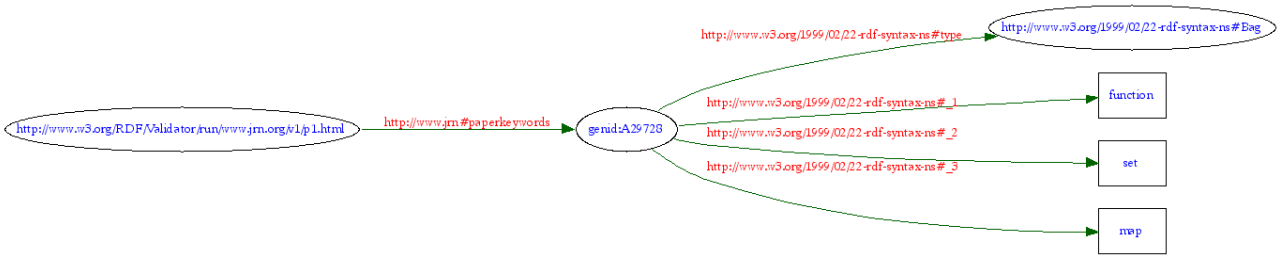
Элемент <rdf:Description> содержит описание ресурса идентифицированного аттрибутом rdf:about.

Ресурс «www.journal.org/volume1/a.html» имеет свойства «journal:author» и «journal:title» (ресурсы), которые принимают соответственно значения: литерал «J. Smith» и «Something».

Проверить, отвечает ли созданный RDF-файл установленным правилам, можно с помощью валидатора RDF (http://www.w3.org/RDF/Validator/). Помимо проверки валидатор записывает модель данных в виде последовательности триплетов и строит RDF-граф.



Number	Subject	Predicate	Object
1	http://www.w3.org/RDF/Validator/run/www.journal.org/volume1/a.html	http://www.journal#author	"J. Smith"
2	http://www.w3.org/RDF/Validator/run/www.journal.org/volume1/a.html	http://www.journal#title	"Something"
3	http://www.w3.org/RDF/Validator/run/www.journal.org/volume1/b.html	http://www.journal#author	"A. Parrot"
4	http://www.w3.org/RDF/Validator/run/www.journal.org/volume1/b.html	http://www.journal#title	"Anything"



Последовательность триплетов, соответствующая вышеприведенному xml-файлу, имеет вид:

Triplet 1
Subject: http://www.w3.org/RDF/Validator/run/www.journal.org/volume1/a.html
Predicate: http://www.journal#author
Object: "J. Smith"
Triplet 2
Subject: http://www.w3.org/RDF/Validator/run/www.journal.org/volume1/a.html
Predicate: http://www.journal#title

```
Object: "Something"
Triplet 3
Subject: http://www.w3.org/RDF/Validator/run/www.journal.org/volume1/b.html
Predicate: http://www.journal#author
Object: "A. Parrot"
Triplet 4
Subject: http://www.w3.org/RDF/Validator/run/www.journal.org/volume1/b.html
Predicate: http://www.journal#title
Object: "Anything"
```

В предыдущем примере для описания RDF-графа мы использовали XML-синтаксис для RDF (http://www.w3.org/TR/rdf-syntax-grammar/). Опишем этот синтаксис подробнее.

7.2. Основные элементы RDF/XML

<rdf:RDF>

Это корневой элемент, определяющий то, что данный XML-документ является RDF-документом. Он также содержит ссылку на пространство имен RDF.

Синтаксис:

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
</rdf:RDF>
```

<rdf:Description>

Этот элемент описывает набор триплетов с данным субъектом. Субъект триплета (ресурс) идентифицируется с помощью атрибута «about». Предикат триплета описывается с помощью тега, а объект триплета описывается как содержимое соответствующего тега.

Синтаксис:

```
<rdf:Description rdf:about="URI">
<PROPERTY_1> OBJECT_1</PROPERTY_1>
....
<PROPERTY_1> OBJECT_1</PROPERTY_1>
</rdf:Description>
```

Это описание эквивалентно набору триплетов

```
(URI,PROPERTY_1,OBJECT_1)
...
(URI,PROPERTY_N,OBJECT_N)
```

В следующем примере

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:journal="http://www.jrn#">
<rdf:Description
rdf:about="www.jrn.org/v1/p1.html">
<journal:author>J. Smith</journal:author>
<journal:title>Something</journal:title>
</rdf:Description>
</rdf:RDF>
```

объектом является ресурс, идентицированный URI «www.jrn.org/v1/p1.html», предикатами – journal:author, journal:title, а объектами – J. Smith и Something. Обратим внимание, что предикаты задаются URI из пространства имен «journal="http://www.journal#». Соответствующие триплеты имеют вид

```
Subject Predicate Object
www.jrn.org/v1/p1.html http://www.jrn#author "J. Smith"
www.jrn.org/v1/p1.html http://www.jrn#title "Something"
```

Синтаксис RDF/XML позволяет описать тот же набор триплетов с помощью аттрибутов, а не элементов. Например, вышеописанный набор триплетов может быть задан как

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:journal="http://www.jrn#">
<rdf:Description
rdf:about="www.jrn.org/v1/p1.html"
journal:author="J. Smith"
journal:title ="Something">
</rdf:Description>
</rdf:RDF>
```

7.3. Дополнительные структуры RDF/XML

В RDF предусмотрен ряд способов группировки данных.

Пустой узел – это узел графа RDF, который не содержит данных, но используется как родительский узел для группировки данных. Группировку данных можно пояснить на следующем примере.

Дата выхода тома журнала состоит из номера месяца и номера года. Допустим, что нам для обработки данных нужно иметь свойство «дата» и подсвойства «месяц» и «год». Это реализуется следующим образом.

Вводится пустой узел, определяемый URI genid:VOLUMEDATE (этот URI произволен) и записываются тройки

```
Subject Predicate Object
www.jrn.org/v1.html http://www.jrn#date genid:VOLUMEDATE
genid:VOLUMEDATE http://www.jrn#datemonth "03"
genid:VOLUMEDATE http://www.jrn#dateyear "2007"
```

В нотации RDF/XML пустой узел описывается элементом <rdf:Description> без аттрибута about. Вышеописанный пример записывается следующим образом:

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:journal="http://www.jrn#">
<rdf:Description>
<journal:datemonth>03</journal:datemonth>
<journal:dateyear>2008</journal:dateyear>
</rdf:Description>
</rdf:RDF>
```

Контейнеры

Элемент <rdf:Bag> используется для того, чтобы описать неупорядоченный список элементов. Этот элемент может содержать повторяющиеся значения. В следующем примере описываются ключевые слова статьи.

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:journal="http://www.jrn#">
<rdf:Description
rdf:about="www.jrn.org/v1/p1.html">
<journal:paperkeywords>
<rdf:Bag>
<rdf:li>function</rdf:li>
<rdf:li>set</rdf:li>
<rdf:li>map</rdf:li>
</rdf:Bag>
</journal:paperkeywords>
</rdf:Description>
</rdf:RDF>
```

Укажем набор триплетов, соответствующий этому набору RDF-данных, который сгенерирован программой валидатором (http://www.w3.org/RDF/

Validator/). Будем записывать триплеты в форме (Subject, Predicate, Object) и нумеровать их.

1. **(genid:A31721, http://www.w3.org/1999/02/22-rdf-syntax-ns#type, http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag)**
2. **(http://www.jrn.org/v1/p1.html, http://www.jrn#paperkeywords, genid:A31721)**
3. **(genid:A31721, http://www.w3.org/1999/02/22-rdf-syntax-ns#_1, "function")**
4. **(genid:A31721, http://www.w3.org/1999/02/22-rdf-syntax-ns#_2,"set")**
5. **(genid:A31721, http://www.w3.org/1999/02/22-rdf-syntax-ns#_3, "map")**

Для представления элемента Bag в виде триплетов используется ресурс, определяемый URI genid:A31721 (его имя автоматически генерируется программой). Он является родительским узлом для узлов, входящих в список.

Элемент <rdf:Seq>. Этот элемент описывает упорядоченный список свойств и может содержать повторяющиеся значения. Например,

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:journal="http://www.jrn#">
<rdf:Description
rdf:about="www.jrn.org">
<journal:auhors>
<rdf:Seq>
<rdf:li>J. Smith</rdf:li>
<rdf:li>M. Johnson</rdf:li>
<rdf:li>S. Bush</rdf:li>
</rdf:Seq>
</journal:authors>
</rdf:Description>
</rdf:RDF>
```

Укажем набор триплетов, соответствующий этому набору RDF-данных. Будем записывать триплеты в форме (Subject, Predicate, Object) и нумеровать их.

1. **(genid:A31880, http://www.w3.org/1999/02/22-rdf-syntax-ns#type, http://www.w3.org/1999/02/22-rdf-syntax-ns#Seq)**
2. **(http://www.jrn.org, http://www.jrn#authors, genid:A31880)**
3. **(genid:A31880, http://www.w3.org/1999/02/22-rdf-syntax-ns#_1, "J. Smith")**
4. **(genid:A31880, http://www.w3.org/1999/02/22-rdf-syntax-ns#_2 ,"M. Johnson")**
5. **(genid:A31880, http://www.w3.org/1999/02/22-rdf-syntax-ns#_3, "S. Bush")**

Элемент <rdf:Alt>. Этот элемент предназначен для описания списка, из которого пользователь может выбрать только одно значение:

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
```

```
xmlns:journal="http://www.jrn#">
<rdf:Description
rdf:about="www.jrn.org">
<journal:papers>
<rdf:Alt>
<rdf:li>www.jrn.org/v1</rdf:li>
<rdf:li>www.jrn.org/v2</rdf:li>
<rdf:li>www.jrn.org/v3</rdf:li>
</rdf:Alt>
</journal:papers>
</rdf:Description>
</rdf:RDF>
```

1. **(genid:A31924, http://www.w3.org/1999/02/22-rdf-syntax-ns#type, http://www.w3.org/1999/02/22-rdf-syntax-ns#Alt)**
2. **(http://www.jrn.org, http://www.jrn#papers, genid:A31924)**
3. **(genid:A31924, http://www.w3.org/1999/02/22-rdf-syntax-ns#_1, "www.jrn.org/v1")**
4. **(genid:A31924, http://www.w3.org/1999/02/22-rdf-syntax-ns#_2, "www.jrn.org/v2")**
5. **(genid:A31924, http://www.w3.org/1999/02/22-rdf-syntax-ns#_3, "www.jrn.org/v3")**

Наборы

В элементах, задающих контейнеры, не предполагается, что контейнер описывает группу, содержащую только данные элементы. Набор отличается от контейнера тем, что он описывает группу, содержащую только указанные элементы.

Пример 7.3.1. Этот набор RDF данных описывает журнал, в котором ровно три номера.

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:journal="http://www.jrn#">

<rdf:Description
rdf:about="www.jrn.org">
<journal:volumes rdf:parseType="Collection">
<rdf:Description rdf:about="http://www.jrn.org/v1"/>
<rdf:Description rdf:about="http://www.jrn.org/v2"/>
<rdf:Description rdf:about="http://www.jrn.org/v3"/>
</journal:volumes>
</rdf:Description>
```

```
</rdf:RDF>
```

1. **(http://www.jrn.org, http://www.jrn#volumes, genid:A32230)**
2. **(genid:A32230, http://www.w3.org/1999/02/22-rdf-syntax-ns#first, http://www.jrn.org/v1),**
3. **(genid:A32230, http://www.w3.org/1999/02/22-rdf-syntax-ns#rest, genid:A32231)**
4. **(genid:A32231, http://www.w3.org/1999/02/22-rdf-syntax-ns#first, http://www.jrn.org/v2)**
5. **(genid:A32231, http://www.w3.org/1999/02/22-rdf-syntax-ns#rest, genid:A32232)**
6. **(genid:A32232, http://www.w3.org/1999/02/22-rdf-syntax-ns#first, http://www.jrn.org/v3)**
7. **(genid:A32232, http://www.w3.org/1999/02/22-rdf-syntax-ns#rest, http://www.w3.org/1999/02/22-rdf-syntax-ns#nil)**

7.4. Dublin Core в терминах RDF

Среду RDF можно рассматривать как среду метаданных (данных о данных). В 1995 году на Metadata Workshop в городе Дублин штата Огайо был выделен набор метаданных – Dublin Core Metadata Initiative (DCMI) [28], описывающий ряд предопределенных свойств документов. Приведем этот набор в виде следующей таблицы (http://dublincore.org/documents/2006/12/18/dces/; перевод Алексея Бешенова с сайта http://beshenov.ru)

Элементы

Элемент: contributor

URI:	http://purl.org/dc/elements/1.1/contributor
Метка:	*Участник*
Определение:	Сущность, участвующая в создании содержимого ресурса.
Комментарий:	Человек, организация или сервис; обычно совпадает с именем человека, названием организации или сервиса.

Элемент: coverage

URI:	http://purl.org/dc/elements/1.1/coverage
Метка:	*Охват*
Определение:	Пространство или границы, с которыми связано содержимое ресурса.
Комментарий:	Как правило, географическое положение (название места или координаты), временной период (название периода, дата, набор дат) или подведомственная область (такая как административная область). Рекомендуется выбирать значение из определенного словаря (например, из «Thesaurus of Geographic Names») и использовать, где это возможно, именованные места и периоды времени вместо числовых идентификаторов (координат или промежутков дат).

Элемент: creator

URI:	http://purl.org/dc/elements/1.1/creator
Метка:	*Автор*
Определение:	Сущность, ответственная за производство содержимого ресурса.
Комментарий:	Человек, организация или сервис; обычно совпадает с именем человека, названием организации или сервиса.

Элемент: date

URI:	http://purl.org/dc/elements/1.1/date
Метка:	*Дата*
Определение:	Дата события в жизненном цикле ресурса.
Комментарий:	Может использоваться для выражения временной информации на любом уровне точности. Рекомендуется представлять дату в соответствии с ISO 8601.

Элемент: description

URI:	http://purl.org/dc/elements/1.1/description
Метка:	*Описание*
Определение:	Описание содержимого ресурса.
Комментарий:	Резюме, оглавление, ссылку на графическое представление контента, произвольное описание и другую подобную информацию.

Элемент: format

URI:	http://purl.org/dc/elements/1.1/format
Метка:	*Формат*
Определение:	Физическое или цифровое представление ресурса, измерение.
Комментарий:	Измерение может быть, например, размером или продолжительностью. Рекомендуется выбирать значение из определенного словаря, например, списка типов данных Интернет MIME.

Элемент: identifier

URI:	http://purl.org/dc/elements/1.1/identifier
Метка:	*Идентификатор*
Определение:	Конкретная ссылка на ресурс в данном контексте.
Комментарий:	Рекомендуется определять ресурс при помощи строки или числа, удовлетворяющего формальной системе идентификации.

Элемент: language

URI:	http://purl.org/dc/elements/1.1/language
Метка:	*Язык*
Определение:	Национальный язык содержимого
Комментарий:	Рекомендуется выбирать значение из определенного словаря, такого как RFC 3066.

Элемент: publisher

URI:	http://purl.org/dc/elements/1.1/publisher
Метка:	*Издатель*
Определение:	Сущность, делающая ресурс доступным.
Комментарий:	Человека, организация или сервис; обычно совпадает с именем человека,

названием организации или сервиса.

Элемент: relation

URI:	http://purl.org/dc/elements/1.1/relation
Метка:	*Связь*
Определение:	Ссылка на связанный ресурс.
Комментарий:	Рекомендуется определять *Связь* при помощи строки или числа, удовлетворяющего формальной системе идентификации.

Элемент: rights

URI:	http://purl.org/dc/elements/1.1/rights
Метка:	*Правовая информация*
Определение:	Правовая информация, связанная с ресурсом.
Комментарий:	*Правовая информация* содержит правовые соглашения касательно ресурса, включая информацию о правах на интеллектуальную собственность.

Элемент: source

URI:	http://purl.org/dc/elements/1.1/source
Метка:	*Источник*
Определение:	Ссылка на ресурс, на основе которого составлен данный ресурс.
Комментарий:	Данный ресурс может быть составленным из *Источника* частично или полностью. Рекомендуется определять *Источник* при помощи строки или числа, удовлетворяющего формальной системе идентификации.

Элемент: subject

URI:	http://purl.org/dc/elements/1.1/subject
Метка:	*Тема*
Определение:	Тема содержимого ресурса.
Комментарий:	Как правило, выражается ключевыми словами, фразами, либо кодами классификации. Рекомендуется выбирать значение из определенного словаря. Пространственная или временная принадлежность ресурса должна описываться элементом coverage.

Элемент: title

URI:	http://purl.org/dc/elements/1.1/title
Метка:	*Заголовок*
Определение:	Название ресурса.
Комментарий:	Как правило, совпадает с названием, под которым формально известен ресурс.

Элемент: type

URI:	http://purl.org/dc/elements/1.1/type
Метка:	*Тип*
Определение:	Вид или жанр содержимого ресурса.
Комментарий:	Рекомендуется выбирать значение из определенного словаря, такого как DCMI Type Vocabulary. Физическое или цифровое представление ресурса определяется элементом format.

Приведем пример набора метаданных Dublin Core для электронного математического журнала Lobachevskii Journal of Mathematics, описанного с помощью RDF:

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc= "http://purl.org/dc/elements/1.1/">
<rdf:Description rdf:about="http://www.jrn.org">
<dc:title>Lobachevskii Journal of Mathematics</dc:title>
<dc:description>Electronic Mathematical Journal</dc:description>
<dc:publisher>Kazan State University</dc:publisher>
<dc:date>1996-08-02</dc:date>
<dc:type>Text</dc:type>
<dc:format>text/html</dc:format>
<dc:language>en</dc:language>
</rdf:Description>
</rdf:RDF>
```

8. Метаданные в XML-документах

К сожалению, сейчас невозможно вставить полный RDF в XHTML без нарушения валидности результирующего XHTML, за исключением использования элементов meta и link в заголовке. Наилучшим решением будет сохранить RDF отдельно и использовать URI для того, чтобы сослаться на страницу XHTML, и элемент link в странице XHTML для ссылки на содержимое RDF. Эта техника часто называется RDF autodiscovery link и уже используется целым рядом инструментов. Однако работа по улучшению интеграции RDF в документы продолжается.

Рабочая Группа GRDDL недавно разработала шлюз для данных в микроформатах и Группа по Распространению Semantic Web, работая над RDF, создала модуль XHTML1.1, который позволит использовать фактически любой словарь RDF для аннотирования XHTML-контента, почти как микроформаты, но более точно и с лучшими возможностьями для *интеграции* разных словарей в одном документе. Наконец, eRDF (разработанный Талисом) предлагает промежуточный вариант, когда можно добавлять произвольные данные RDF в станицу (X)HTML без необходимости использовать новый модуль, хотя и с ограничением на типы RDF-словарей, которые могут быть использованы таким образом.

В веб-страницах метаданные используются для обозначения ключевой информации о содержимом данной страницы. Роботы поисковых машин осуществляют выборку, в основном, по информации, представленной в блоке мета-

данных. Поэтому включение метаданных в HTML-файл позволяет определить параметры поиска на этапе создания. Известной рекомендацией по описанию основных характеристик информационного ресурса является Dublin Core Metadata Set (DC) [19].

Пример 8.1. Метаданные используются для описания статьи электронного журнала

```
<HTML XMLNS:M="http://www.w3.org/1998/Math/MathML">
<head>
 <OBJECT ID=article1
    CLASSID="clsid:32F66A20-7614-11D4-BD11-00104BD3F987">
 </OBJECT>
  <?IMPORT NAMESPACE="M" IMPLEMENTATION="#article1">
<!-- Блок метаданных DC: -->
<META content="Splitting iterative methods and parallel solution of variational in-
equalities" name=DC.Title>
<META content="E.Laitinen" name=DC.Author.PersonalName>
<META content="A.Lapin" name=DC.Author.PersonalName.2>
<META content="J.Pieska" name=DC.Author.PersonalName.3>
<META content = 'Splitting iterative methods for the sum of maximal monotone and
single-valued monotone operators in a finite-dimensional space are studied: convergence,
rate of convergence and optimal iterative parameters are derived.' name=DC.Description>
<META content="mathematics, applied mathematics, scientific computing"
name=DC.Subject>
<META content="Lobachevskii Journal of Mathematics" name=DC.Publisher>
<!—Конец блока метаданных DC -->
</head>
<body>
<H2 align=center>E. Laitinen, A.V. Lapin, J.Pieska</H2>
<H2 align=center>Splitting iterative methods and parallel solution of
variational inequalities
</H2>
<H2 align=center>Lobachevskii Journal of Mathematics, </H2>
<H2 align=center>Vol.8, pp.167-184 </H2>
<P>
 Splitting iterative methods for the sum of maximal monotone and  .....
</P>
<!-- MathML Content Markup  -->
<m:math>
<m:mi>a</m:mi>
   <m:msup>
    <m:mi>x</m:mi>
    <m:mn>2</m:mn>
```

```
    </m:msup>
    <m:mo>+</m:mo>
    <m:mi>b</m:mi>
    <m:mi>x</m:mi>
     <m:mo>+</m:mo>
    <m:mi>c</m:mi>
    <m:mo>=</m:mo>
    <m:mn>0</m:mn>
  </m:math>
  <p>
  ...
  </p>
  </body>
  </html>
```

Включить метаданные в XML-файл подобным образом не удастся. Однако на помощь может придти RDF. В следующем примере для MathML-документа, **article.xml**, содержащего статью журнала, дается набор метаданных Dublin Core в формате RDF – документ **article.rdf**.

Пример 8.2.

```
<?xml version="1.0" encoding="WINDOWS-1251"?>
<!-- article.xml -->
 <article>
  <autor>E. Laitinen, A.V. Lapin, J.Pieska</autor>
  <title>Splitting iterative methods and parallel solution of
       variational inequalities
  </title>
  <bibl>Lobachevskii Journal of Mathematics,
       Vol.8, pp.167-184)
  </bibl>
  <abstract>
     Splitting iterative methods for the sum of maximal monotone and  .....
<math>
<mi>a</mi>
    <msup>
      <mi>x</mi>
      <mn>2</mn>
    </msup>
   <mo>+</mo>
   <mi>b</mi>
   <mi>x</mi>
    <mo>+</mo>
   <mi>c</mi>
```

```
    <mo>=</mo>
    <mn>0</mn>
  </math>
   </abstract>
  </article>
```

```
<?xml version="1.0"?>
<!--  article.rdf -->
<rdf:RDF   xmlns:rdf=
"http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/metadata/dublin_core#">
<rdf:Description rdf:about="http://ljm.ksu.ru/vol8/latin.htm">
    <dc:Publisher>
      Lobachevskii Journal of Mathematics
</dc:Publisher>
    <dc:Title>
Splitting iterative methods and parallel
solution of variational inequalities
</dc:Title>
    <dc:Author>
     <rdf:Seq ID="AutorsAlphabeticalBySurname"
       rdf:_1="E.Laitinen"
       rdf:_2="A.V. Lapin"
       rdf:_3="J.Pieska"/>
  </dc:Author>
   <dc:Subject>
     mathematics,  applied mathematics,
     scientific computing
  </dc:Subject>
  </rdf:Description>
</rdf:RDF>
```

В следующем примере показано, как включить набор метаданных Dublin Core в формате RDF в XML-документ.

Пример 8.3.

```
<?xml version="1.0" encoding="windows-1251"?>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>
<body>
<!--  RDF набор метаданных: -->
<rdf:RDF  xmlns:rdf=
  "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
```

```
  xmlns:dc="http://purl.org/metadata/dublin_core#">
<rdf:Description  rdf:about=
          "http://ljm.ksu.ru/vol8/latin.htm">
   <dc:Publisher>
      Lobachevskii Journal of Mathematics
  </dc:Publisher>
   <dc:Title>
Splitting iterative methods and parallel
solution of variational inequalities
</dc:Title>
    <dc:Author>
     <rdf:Seq ID="AutorsAlphabeticalBySurname"
       rdf:_1="E.Laitinen"
       rdf:_2="A.V. Lapin"
       rdf:_3="J.Pieska"/>
  </dc:Author>
    <dc:Subject>mathematics,  applied mathematics, scientific computing</dc:Subject>
  </rdf:Description>
</rdf:RDF>

<p> Splitting iterative methods for the sum of maximal monotone and  .....
<!-- MathML вставка -->
<math xmlns="http://www.w3.org/1998/Math/MathML">
  <mi>a</mi>
    <msup>
      <mi>x</mi>
      <mn>2</mn>
    </msup>
   <mo>+</mo>
   <mi>b</mi>
   <mi>x</mi>
    <mo>+</mo>
   <mi>c</mi>
   <mo>=</mo>
   <mn>0</mn>
</math>
</p>
</body>
</html>
```

Литература

1. *Berners-Lee T.* Semantic Web Road map. http://www.w3.org/DesignIssues/ Semantic.html; Рус. перевод: http://gridclub.ru/library/publication. 2007-04-23.2195467714/view.
2. W3C Semantic Web Activity Statement, http://www.w3.org/2001/sw/Activity
3. *Hendler J.* Agents and the Semantic Web// IEEE Intelligent Systems Journal, March/April 2001. – V. 16, No 2. – P. 30-37.
4. *Паринов С.* Онлайновая революция в науке начинается//Соционет. – http://sparinov. socionet.ru/files/online-future-science-full.doc.ru.
5. *Семантическая паутина*. – Электронная энциклопедия Википедия. – http://ru. wikipedia.org/wiki/.
6. Стренталл Д. Третий Веб. – http://www.xakep.ru/post/40176/.
7. *Рапоза Д.* Плетение "семантической паутины" // PC Week. – 2007. – № 22.
8. *Berners-Lee T., Hendler J., Lassila O.* The Semantic Web // Scientific American, May 17, 2001; Русский перевод: Семантическая Сеть – http://ezolin.pisem.net/logic/semantic_web_rus. html.
9. Язык онтологий в Web. – http://wmast.com.ua/article.php.
10. Berners-Lee T. Semantic Web on XML. – http://www.w3.org/2000/Talks/ 1206-xml2k-tbl/slide1-0.html
11. *Buneman P., Khanna S., Tajima K., Tan W-C.* Archiving Scientific Data, SIG-MOD Conference 2002.
12. *Hendler J., Parsia B*. XML and the Semantic Web
13. *Extensible Markup Language (XML) 1.0 (Second Edition) W3C Recommendation*. – http://www.w3.org/TR/2000/REC-xml-20001006; Русский перевод: http://www.rol.ru/news/it/ helpdesk/xml01.htm).
14. *Semantic Web: роли XML и RDF* // Открытые системы. – 2001. – № 9. – http://osp.admin.tomsk.ru/os/2001/09/041.htm.
15. *Питц-Моултис Н., Кирк Ч.* XML – СПБ: БХВ – Петербург, 2000. – 736 с.
16. *Хабибуллин И. Ш.* Самоучитель XML. – СПБ: БХВ – Петербург, 2003. – 336 с.
17. *Extensible Stylesheet Language (XSL) Version 1.1 W3C Recommendation* 05 December 2006 – http://www.w3.org/TR/xsl/
18. *XSL Transformations (XSLT) Version 1.0 W3C Recommendation* 16 November 1999 http://www.w3.org/TR/1999/REC-xslt-19991116; Рус. перевод: http://www.rol.ru/news/it/ helpdesk/xslt01.htm.
19. *XSL Transformations (XSLT) Version 2.0. W3C Recommendation* 23 January 2007 – http://www.w3.org/TR/xslt20.
20. *XML Path Language (Xpath) Version 1.0 W3C Recommendation* 16 November 1999 – http://www.w3.org/TR/xpath.
21. *Miloslav Nic, Jiri Jirat*, XPath Tutorial, http://www.zvon.org/xxl/XpathTutorial/ General/examples.html; Рус. перевод: – http://www.zvon.org/xxl/XPathTutorial/ General_rus/ examples.html

22. XPath Tutorial, – http://www.w3schools.com/xpath/.
23. *Школы консорциума W3C,* – www.xml.nsu.ru.
24. *Mathematical Markup Language (MathML) Version 2.0 (Second Edition) W3C Recommendation* 21 October 2003 – http://www.w3.org/TR/MathML2/
25. The Unicode Character Code Charts By Script, http://www.unicode.org/charts/
26. *Pankaj Kamthan.* XML Namespaces: Universal Identification in XML Markup. – http://tech.irt.org/articles/js193/.
27. *Resource Description Framework (RDF) /* W3C Semantic Web Activity. – http://www.w3.org/RDF/.
28. *Dublin Core Metadata Element Set.* – http://purl.org/metadata/ dublin_core_ elements.

Полезные ссылки

1. *Питц-Моултис Н., Кирк Ч.* XML – СПБ: БХВ – Петербург, 2000. – 736 с.
2. *Bray T, Dave Hollander D., Layman A. (Editors).* Namespaces in XML (W3C Recommendation). – http://www.w3.org/TR/REC-xml-names/.
3. *Гуссенс М., Рапц С.* Путеводитель по пакету LaTeX и его Web-приложениям. – М.: Мир, 2001. – с.
4. *An XHTML + MathML + SVG Profile.* – http://www.w3.org/TR/2002/WD-XHTMLplus-MathMLplusSVG-20020430/xhtml-math-svg.html.
5. *Carlisle D.* MathML Files: DSSSL style sheet for MathML. – http://www.nag.co.uk/ projects/OpenMath/mml-files/.
6. *Gentle Introduction to MathML.* – http://www.dessci.com/en/support/tutorials/ mathml/default.htm.
7. *History of MathML. The Two Threads of Mathematical Computer Languages.* – http://www.mathmlcentral.com/history.html.
8. *MathML – What's in it for us.* – http://tech.irt.org/articles/js081/.
9. *MathPlayer. Display MathML in your browser.* – http://www.dessci.com/en/products/ mathplayer/.
10. *OpenMath and MathML.* – http://www.openmath.org/cocoon/openmath//projects/ esprit/final/node6.htm.
11. *OpenMath and MathML: Semantic Mark Up for Mathematics.* – http://www.acm.org/crossroads/xrds6-2/openmath.html.
12. *OpenMath: Accessing and Using Mathematical Information Electronically.* – http://www.nag.co.uk/projects/OpenMath/.
13. *Putting mathematics on the Web with MathML.* – http://www.w3.org/ Math/XSL/.
14. *Strategies for Math on the Web.* – http://www.dessci.com/en/reference/webmath/ strategies.htm.
15. *MathML for Math and Science Communication.* – http://www.dessci.com/en/reference/ webmath/tech/mathml.htm.
16. *Math Typesetting for the Internet*. – http://mathforum.org/typesetting/ index.html.
17. *Wolfram Research Contributes Central Ideas to Web Math Standard.* – http://www.wolfram.com/news/archive/mathml.html.
18. *Stephen Wolfram.* Mathematical Notation: Past and Future. – http://www. stephen-wolfram.com/publications/talks/mathml/.
19. *Kohlhase M.* MathML Presenting and Capturing Mathematics for the Web. – http://www.w3.org/Math/Documents/mathml-tutorial.pdf.
20. *W3C MathML 2.0 Specification.* – http://www.w3.org/Math (частичный рус. перевод – на http://www.raleigh.ru/MathML/MathML2/).
21. *W3C MathML 1.0 Specification.* – http://www.w3.org/TR/REC-MathML-19980407.
22. *Ширки К.* Семантический Веб, Силлогизм и представления о мире. – http://www.shirky.com/writings/semantic_syllogism.html; Рус. перевод: http://www. se-

mantictools.ru/conception/semantic_syllogism.shtml/

23. *Доктороу К.* Метачушь: проливаем свет на семь логических несуразиц по поводу мета-утопии. – http://www.well.com/~doctorow/metacrap.htm; Рус. перевод: http://www.semantictools.ru/conception/metacrap.shtml.
24. *Chapmin P.-A.* RDF Tutorial, http://bat710.univ-lyon1.fr/~champin/rdf-tutorial/.
25. *Eva Mª Méndez Rodríguez* RDF: un modelo de metadatos flexible para las bibliotecas digitales del próximo milenio/http://www.bib.uc3m.es/~mendez/publicaciones/7jc99/rdf.htm.
26. *Tim Bray.* What is RDF? http://www.xml.com/pub/a/2001/01/24/rdf1.html.
27. *Expressing Simple Dublin Core in RDF/XML*, http://dublincore.org/documents/dcmes-xml/.
28. *Expressing Qualified Dublin Core in RDF/XML*, http://dublincore.org/documents/dcq-rdf-xml/.
29. *Stefan Decker, Frank van Harmelen, Jeen Broekstra, Michael Erdmann , Dieter Fensel , Ian Horrocks, Michel Klein , Sergey Melnik*, The Semantic Web - on the respective Roles of XML and RDF, http://www.ontoknowledge.org/oil/downl/IEEE00.pdf.
30. *Berners-Lee T.* Semantic Web on XML. – http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide1-0.html.
31. Язык онтологий в Web. – http://wmast.com.ua/article.php.
32. *Van der Vlist T.* XML Schema. – O'Reily, 2002. – 400 p.
33. *Когаловский М.Р.* XML: сферы применения //Директор информационной службы. Апрель 2001. – С. 10-12.
34. *Когаловский М.Р.* Стандарты XML и электронные библиотеки // Электронные библиотеки. – 2003. – Т. 6, Вып. 2.
35. *Когаловский М.Р.* Тенденции развития технологий управления информационными ресурсами //Тр. 8ой Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» – RCDL'2006, Суздаль, Россия, 2006. – С. 46-55.
36. *Когаловский М.Р.* Стандарты платформы XML и базы данных. Обзорный доклад. Сб. тр. Третьей Всероссийской конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции», Петрозаводск, 11 – 13 сент. 2001 г. – Петрозаводск: Карельский научный центр РАН, 2001. –- С. 108-117.
37. *Herman I.* Questions (and Answers) on the Semantic Web // XML – Days, Berlin, Germany, 2006-09-20.
38. *Palmer S. B.* The Semantic Web: An Introduction. –http://infomest.net/2001/swintro; Русский перевод: http://gridclub.ru/library/publication.2007-01-23.2882229210/view/.
39. *Swartz A.* with assistance. The Semantic Web In Breadth. – http://logicerror.com/ semanticWeb-long; Русский перевод: http://gridclub.ru/library/publication. 2006-12-04.7455713114/view/.
40. Carole Goble? David De Roure. The Grid: An Application of the Semantic Web. – http://www.semanticgrid.org/documents/sigmod/ami9.pdf; Рус. перевод: http://grid-

club.ru/ library/publication.2007-04-05.7789683187/view.
41. *Passin T.B.* Explorer's Guide to the Semantic Web. – Manning Publ., 2004. – 282 p.

А. М. Елизаров, Е. К. Липачёв, М. А. Малахальцев

ОСНОВЫ MathML.
Представление математических текстов в Internet

Практическое руководство, 2-е издание

Изд. лиц. № 0243 от 20.01.99. Подписано в печать 16.11.08
Бумага офсетная. Формат 60x90 1/16. Гарнитура «Таймс».
Тираж 100 экз., объем – 100 с.
Заказ . Печать ризографическая

Издательство Казанского математического общества
420008, Казань, Ул. Профессора Нужина, 1/37