Тема моего доклада это расширение DTD-document type definition, что в переводе с английского определение типа документа. Я расскажу вам его назначение, правила использования и синтаксис.

Все мы знаем, что XMl документы можно разделить на 2 группы, **корректные** (well-formed), удовлетворяющими определенному набору требований; **действительные** (valid), т.е. соответствующие всем требованиям, предъявляемым к корректным XML документам, и удовлетворяющие условиям, подключенной к ним схемы. С проверкой документа на корректность проблем не возникает: если ошибок не выскочило и всё отобразилось так, как мы хотели, то документ корректен. Например, если в HTML-документе написать что-то вроде «<Z>Привет!</Z>», то наш документ будет полностью корректен, но проигнорирован браузером. Почему? Потому что браузер ничего не знает о том, что это за «Z» такой. И если мы проверим наш документ на допустимость с помощью валидатора (валидатор - это программное обеспечение, выполняющее проверку кода на соответствие общепринятым стандартам консорциума), то документ таковым признан не будет. А как об этом узнает валидатор и на основании чего он вынес такой вердикт?

В настоящее время World Wide Web Consortium (W3C) поддерживает две схемы: DTD (Document Type Definition) и XML Schema.

Допустимость проверяется с помощью определения типа документа (DTD, document type definition). DTD может быть описан как внутри документа, так и вынесен в отдельный файл (аналогия с CSS: встроенные и подключаемые таблицы стилей). Описание DTD выполнено в Рекомендации «Extensible Markup Language (XML) 1.0», т.е. в той же самой рекомендации, где описаны требования к XML документу.

Задача DTD определить:

* состав элементов, которые могут использоваться в XML документе;
* описание моделей содержания, т.е. правил вхождения одних элементов в другие;
* состав атрибутов, с какими элементами XML документа они могут использоваться;
* каким образом атрибуты могут применяться в элементах;
* описание сущностей, включаемых в XML документ.

DTD, относительно XML документа, могут быть внутренними и внешними.

Также возможно вXML документе одновременно использовать оба типа DTD.

Для объявления DTD в XML документе используется предложение **DOCTYPE**, которое должно располагаться перед корневым элементом XML документа, а указываемое в нем имя DTDдолжно совпадать с именем корневого элемента. Предложение DOCTYPE имеет вид: (см.на слайд)

Внешний DTD находится в отдельном файле, который должен иметь расширение **.dtd**, например, library.dtd. Синтаксис предложения DOCTYPE в случае внешнего подключения принимает следующий вид: (см.на слайд)

SYSTEM и PUBLIC – альтернативно используемые ключевые слова, применяемые по следующему алгоритму.

При указании SYSTEM анализатор должен найти DTD по явно указанному URI, например:

<!DOCTYPE  library SYSTEM “http://www.library.org/dtd/library.dtd”>

При указании PUBLIC считается, что будет использоваться публичный DTD, «хорошо знакомый» используемому обработчику XML документа. Предположим, что для библиотечных систем существует публичная схема DTD, расположенная на определенном сайте в Интернет. Если для работы с библиотечными XML документами используется не стандартный, а специализированный анализатор, то он может сам знать , где находится необходимая ему схема DTD. В этом случае не обязательно указывать точное URI расположения файла со схемой DTD, например:

<!DOCTYPE  library PUBLIC “world/library.dtd”>

Можно совместить описание РUBLIC с указанием конкретного расположения файла со схемой DTD. В этом случае, если анализатор не найдет публичного схему, он будет использовать схему, указанную по конкретному адресу, например:

<!DOCTYPE  library PUBLIC "world/library.dtd"

"http://www.library.org/dtd/library.dtd">

Внутренний DTD включается непосредственно в тело XML документа. Предложение DOCTYPE в случае внутреннего подключения имеет следующий вид: (см.на слайд)

Содержание DTD располагается внутри квадратных скобок.

**1.3 Одновременное использование внешнего и внутреннего DTD**

При использовании внешнего DTD может возникнуть ситуация, когда какие-либо его описания могут не удовлетворять. Например, в XML документе необходимо использовать несколько новых элементов или изменить тип ряда атрибутов. Непосредственно корректировать внешний DTD достаточно нельзя, так как его могут использовать другие программы, и можно нарушить правильность их работы.

В такой ситуации рекомендуется использовать внешний DTD, а необходимые изменения выполнить во внутреннем DTD.

Примечание. В случае возникновения конфликта между описаниями внешнего и внутреннего DTD приоритет имеют описания внутреннего DTD.

Синтаксис предложения DOCTYPE при одновременном использовании внешнего и внутреннего DTD имеет следующий вид: (см.на слайд)

XML документ состоит из набора тегов. С точки зрения DTD все теги, представленные в XML документе являются элементами и должны быть соответствующим образом описаны.

Теги XML документа могут быть двух типов:

·       контейнерные, представляются парой тегов (начальный и конечный);

·       пустые, представляются одиночным тегом. Признаком пустого тега является слеш перед закрывающейся угловой скобкой.

Контейнерный тег хранит в себе текст или другие теги. В примере, описывающем библиотечный каталог, все теги контейнерные. Так тег <book> является контейнерным, поскольку хранит в себе другие теги <title> и <author>. Соответственно теги <title> и <author> являются контейнерными, поскольку хранят в себе текст: название книги и имя автора.

Пустые теги не могут содержать в себе текст или другие теги. Обычно они используются для включения объектов. Что такое объекты, и какие виды объектов существуют, описывается в разделе «Объекты» данного документа. Здесь, в качестве примера, покажем, как можно включить в текст XML документа такой объект, как графический файл:

**2.2. Объявление элементов**

Элементы в DTD объявляются с помощью предложения, которое имеет следующий формат:

Для XML документа, описывающего библиотечного каталог, фрагмент DTD, в котором объявляются элементы будет иметь следующий вид:

*Примечание.* Данный фрагмент объявления включает описание элементов без учета правил вложенности. Описание этих правил представлено в документе «Описание контейнерной структурыXML документов», который находится по адресу homepages.tversu.ru/~berman/dtd/cont/conteiner.htm

**3.1. Общие понятия атрибутов** Элементы описывают основные понятия предметной области. Атрибут уточняет информацию об этом понятии. Он позволяет понять, какая информация связана с элементами и, самое главное, контролировать тип этой информации.Например, возьмем элемент IMG из языка HTML, который предназначен для задания вывода изображения браузером на экран. (В первой лекции мы говорили, что HTML – это подмножество XML.)В примере элемент IMG имеет шесть атрибутов, которые уточняют различные понятия, связанные с выводом изображения.Следует иметь в виду, что то же самое можно описать и без использования атрибутов, с помощью одних элементов. (см.на слайд) Преимущество использования атрибутов состоит в том, как мы увидим далее, что они позволяют контролировать свои значения, чего нельзя достичь, используя только элементы.**3.2. Типы атрибутов**Существует три типа атрибутов: (см.на слайд)**Обязательные атрибуты** (уже видно из их названия) – это атрибуты, которые всегда должны быть определены для данного типа элемента. В противном случае анализатор будет считать документ не состоятельным и выдаст ошибку во время его синтаксического анализа.**Фиксированные атрибуты** – это такие атрибуты значения которых разработчик XML документа не может изменять. Для таких атрибутов можно использовать либо конкретно указанное значение, либо одно из возможных значений. Фиксированные атрибуты могут иметь значение по умолчанию. В этом случае их в тегах XML документа указывать не обязательно, автоматически будут приняты значения по умолчанию.**Неявные атрибуты** – это не обязательные и не фиксированные, т.е. все атрибуты, которые не подходят под определение обязательных и фиксированных атрибутов. Неявные атрибуты не имеют значения по умолчанию и не обязаны всегда появляться вместе с элементом к которому они относятся.**Определение типов атрибутов.** В DTD типы атрибутов задаются специальными идентификаторами, значение которых представлены в таблице 1.**3.3. Типы значений атрибутов** (см.на слайд )**Предопределенное значение** – это множество заранее определенных значений только одно, из которых может выбрать разработчик XML документа для присвоения атрибуту. Одно из значений множества может использоваться как значение по умолчанию.Например, в HTML атрибут ALIGN, отвечающий за расположение текста на строке может принимать только одно из трех предопределенных ему значений {LEFT, CENTER, RIGHT}.Достоинство использования предопределенных значений атрибута заключается в том, что они дают дополнительный уровень контроля, гарантируя установку правильное значение. **Объект** – это виртуальный блок памяти, который может хранить определенную информацию. До того, как объект будет включен в XML документ, он должен быть объявлен, используя определенное объявление. Более подробно типы объектов и их объявления будут описаны далее в разделе "Объекты".**3.4. Определение атрибутов** (см.на слайд)С одним элементом может связано несколько атрибутов, поэтому все параметры, кроме имя\_элемента, могут повторяться.В предложении ATTLIST тип значения атрибута представлен идентификаторами, значения которых даны в таблице 2.**3.5. Пример описания атрибутов** Допустим, в используемом нами XML документе, есть элемент image, атрибуты которого определяют графический файл и правила его расположения на экране: (см.на слайд) Описание элемента image для нашего примера будет вид: (см.на слайд)Строка 1 описывает элемент image. Параметр empty показывает, что элемент image пустой.Строка 2 задает список атрибутов и описывает уникальный идентификатор ID, который в данном примере является обязательным.Строка 3 описывает атрибут border как неявный, простой текст.Строка 4 описывает атрибут alt как обязательный, простой текст.Строка 5 описывает атрибут src как обязательный, нетекстовый объект.Строка 6 описывает атрибут align как обязательный, предопределенное значение, значение по умолчанию lefr.Строка 7 описывает атрибут valign как фиксированный, простой текст, значение по умолчанию middle. В примере этого атрибута нет, но поскольку он фиксированный, то используется неявно, и представляется своим значением по умолчанию.**4.1. Общие понятия объекты**(см.на слайд)**4.2. Текстовый объект**Текстовый объект - это блок текста, который может часто появляться в тексте документа. В текстовый объект может и входить и фрагмент разметки. Например, в XML документе библиотечного каталога элемент author имеет обязательный атрибут country, который указывает на принадлежность автора к определенной стране. Для того, чтобы сократить запись можно использовать текстовый объект. (см.на слайд)Текстовый объект может быть внутренним, как в приведенном выше примере, и внешним, например: (см.на слайд)Вместо параметра SYSTEM может использоваться параметр PUBLIC. Правила использования параметров SYSTEM и PUBLIC полностью аналогичны правилам их использования в предложении DOCTYPE, которые были описаны ранее, в разделе "Внешние и внутренние DTD".

**4.3. Двоичный объект**(см.на слайд)Признаком двоичного объекта является идентификатор NDATA. Синтаксис объявления двоичного объекта имеет следующий вид: (см.на слайд)Например,

<!ENTITY logo "www.mysite.com/logo/logo.gif"

                           SYSTEM NDATA gif>

Поскольку анализатор может не знать, каким образом или с помощью какой программы обрабатывать двоичный объект, то ему необходимо подсказать. Для этой цели используется предложение NOTATION. Например, если для обработки gif файлов необходимо использовать программу PHOTOSHOP, то предложение NOTATION примет следующий вид:<!NOTATION gif SYSTEM "c:\apps\graphics\photoshop.exe">В том случае, когда анализатор знает как обрабатывать двоичный объект и имеет для этого собственные механизмы обработки, предложение NOTATION задавать не надо.**4.4. Параметрический объект**(см.на слайд)Например, мы хотим ввести в XML документ, описывающий библиотечный каталог, для каждой книги (элемент book) наименование издательства и картинку с логотипом этого издательства.

<book>

          <publish>… </publish>

          <logo src="www.publish.com/logo/logo.gif" border="0"

                      alt="логотип издательства" align="center" />

</book>

Поскольку значения всех атрибутов элементе logo кроме src будут повторяться для всех элементов book, то для упрощения записи можно задать параметрический объект pub\_logo:

<!ENTITY % pub\_logo "

          border CDATA #IMPLIED

          alt CDATA #IMPLIED

          align (left,center,right) #REQUIRED "left"

          valign CDATA #FIXED "middle"

Применив pub\_logo вид элемента logo будет иметь вид:

<logo src="www.publish.com/logo/logo.gif" pub\_logo />

**4.5. Символьные и числовые объекты**(см.на слайд)По умолчанию XML использует кодировку ISO-Latin-1, набор символов которой шире, чем в набор ASCII символов. XML может использовать и другие кодировки. Тип кодировки указывается в командной инструкции xml, с которой всегда начинается XML документ:<?xml version="1.0" encoding="имя кодировки"?>Например, для того, чтобы использовать кириллицу необходимо указать кодировку Windows-1251:<?xml version="1.0" encoding=" Windows-1251"?>Кроме символов, не входящих в набор ASCII, символьные объекты используются для ряда символов, входящих в набор ASCII, но зарезервированных для специальных целей. Например, (см.на слайд)