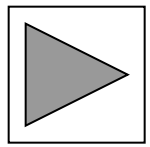
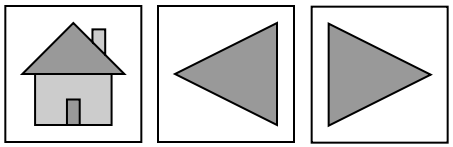
HTML

XML

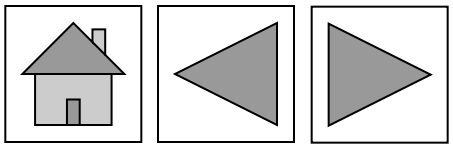




CSS

PHP

Usability



Лекция 2-3

SGML (standard generalized markup language) — HTML, XML

<HTML>
<HEAD>
. . .
</HEAD>
<BODY>
. . .
</BODY>
</HTML>

<HEAD>

<TITLE> → <TITLE> текст <TITLE>

<BASE> → <BASE HREF = “протокол://имя сервера/путь”>

<META> →

<META HTTP_equiv (content, url) – поручает действие серверу

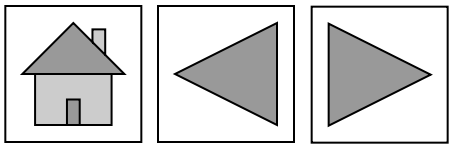
пример

<META http_equiv = “Refresh” content = ‘5
url = http://wasm.ru/index.html
Принудительное обновление страницы каждые 5 сек.

<META http_equiv = “Expires” content = “Data”
Срок годности документа

<META name = “Keywords” lang = “ru” content = “слово1,слово2,слово3…”
<META name = “Description” content “Содержание”

</HEAD>

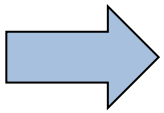


<HTML>
<HEAD>
. . .
</HEAD>
<BODY>
. . .
</BODY>
</HTML>

<BODY>

атрибуты:

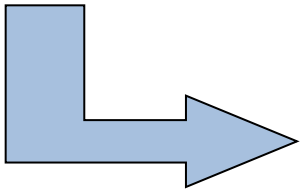
background – фон документа
bgcolor – цвет документа
link – цвет гиперссылки
alink – цвет активной гиперссылки
vlink – цвет посещенной гиперссылки



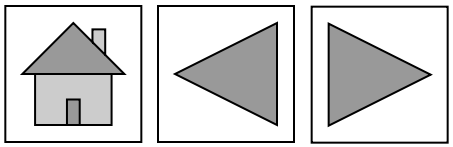
<BODY background="(URL)(путь)имя файла">
<BODY bgcolor="цвет">
<BODY link="цвет">
<BODY alink="цвет">
<BODY vlink="цвет">

topmargin – отступ сверху, right margin – отступ справа
leftmargin – отступ слева, bottommargin - отступ снизу

<address> - идентификация автора



<address> описание
 </address>



Атрибуты тэги выравнивания:

H<номер> - тип заголовка → h1 24p, h2 18p, h3 14p, h4 12p, h5 10p, h6 8p

 - переход на новую строку

<DIV> - выравнивание абзаца

<FONT> - управление внешним видом (FACE (гарнитура), SIZE(размер), COLOR(цвет))

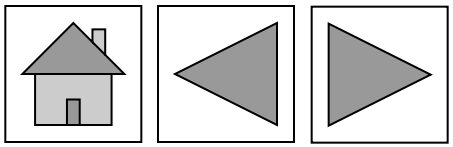
<IMG SRC= . . . > - графическое изображение

<P> - абзац

жирный - - индексируется поиск. системой; <b>

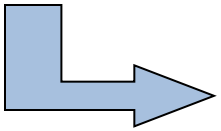
курсив - ; < i>

<u> - подчеркнутый ,   - много пробелов



<HTML>
<HEAD>
. . .
</HEAD>
<BODY>
. . .
</BODY>
</HTML>

Списки:



- <OL> - нумерованный список
- <UL> - маркерованный список
- <MENU> - меню
- <DL> - список определений

<OL type=1 start=1>
<LI> эл.списка<LI>эл.списка </OL>

<UL type=circle>
<LI> эл.списка<LI>эл.списка </UL>

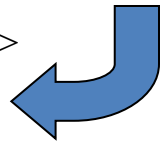
<MENU>
<LI> эл.списка<LI>эл.списка </MENU>

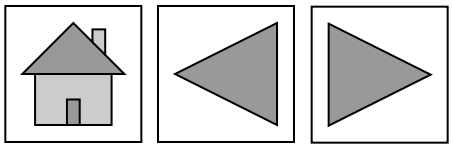
<DL><DT>термин1<DD>опр.1
<DT>термин2<DD>опр.2</DL>

<A> - Ссылки:

- name – метка для перехода внутри текста
  - <A name=имя>текст</A>
- title – визуализацияподсказки
  - <A title="имя ссылки">ссылка</A>
- accesskey – указание горячей клавиши
  - <A accesskey="имя ссылки">ссылки</A>
- href – адрес ссылки
  - <A href="URL">текст</A>
  - <A href="#имя">текст</A>

<A href = "http://www.poshuk.com">Поисковый сервер</A>
<A href = "ftp://ftp.poshuk.com/install.exe">Пример</A>
<A href = mailto:name@domen.ru>Пример</A>





<HTML>
<HEAD>
. . .
</HEAD>
<BODY>
. . .
</BODY>
</HTML>

<TABLE> - Таблица:

атрибуты <TABLE>

- bgcolor -фон ячеек
- background – фоновый рисунок ячеек
- align – горизонтальное выравнивание
- width – ширина таблицы
- структура документа
- valign – верт. выр. → top – вверх; middle – середина; bottom - вниз
- height – высота таблицы (в пикселях)
- cellpadding – отступ внутри ячейки
- cellspasing – расстояние между ячейками
- <sub> - нижний регистр ,<sup> - верхний
- <frame> - отобразить/скрыть линии таблицы
- структура документа → <Tread>; <Tbody>; <Tfoot>

<TABLE align = способ> → способ = left; center; right

<TR> - строка
<CAPTION> - заголовок таблицы
<TH> - заголовок в ячейке
<TD> - ячейка

- rowspan – объединение ячеек столбца -- <TD rowspan = 5>
- colspan – объединение ячеек строки -- <TD colspan = 5>

</BODY>

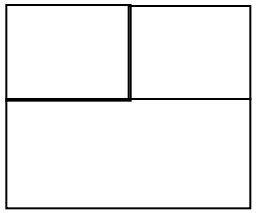
<FRAMESET> - фреймы (кадры)

Заменяет тэг <BODY>

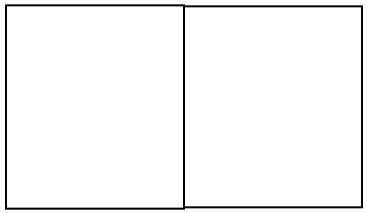
Атрибуты frameset:
cols – количество и размер колонок
rows – количество и размер строк
border – толщина обрамления
frameborder – наличие или отсутствие обрамления

<FRAMESET cols = “число,*,%> cols=50%,50%
<FRAMESET rows = “число,*,%> rows=100,20%,*

<FRAME>
- src – документ фрейма <FRAME scr=“URL”>
- frameborder – обрамление <FRAME frameborder = “1 или 0”
- marginheight – толщина верхнего и нижнего обрамления
- marginwidth – толщина левого и правого обрамления
- name – задает имя фрейма для обращения к нему атрибутом target <A href>
- noresize – лишает возможности изменения размеров фрейма
- scrolling – задает наличие полосы прокрутки (yes, no, auto)



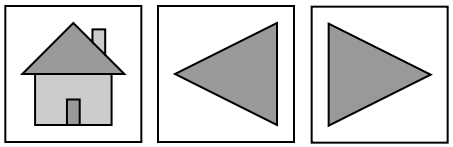
```
<frameset>
<frameset>
  <frame>
  <frame>
</frameset>
  <frame>
</frameset>
```



```
<frameset>
  <frame>
  <frame>
</frameset>
```

Контейнер
Index.htm
  p1.htm – фрейм 1
  p2.htm – фрейм 2

Пример



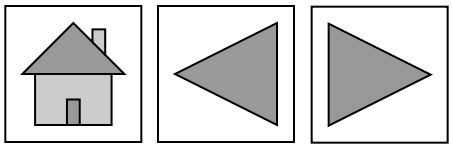
Пример

```
<HTML>
<HEAD>
<TITLE>FRAMES</TITLE>
</HEAD>
<FRAMESET ROWS=150,*>
<FRAME SRC="page1.html" NAME="1">
<FRAMESET COLS=150,*>
  <FRAME SRC="page2.html" NAME="2">
  <FRAME SRC="page3.html" NAME="3">
</FRAMESET>
</FRAMESET>
</HTML>
```

Файл page2.html

```
<HTML>
<HEAD>
<TITLE>page2</TITLE>
</HEAD>
<BODY>
<A HREF="page1.html" target="3"> Link</A>
</BODY>
</HTML>
```

Назад

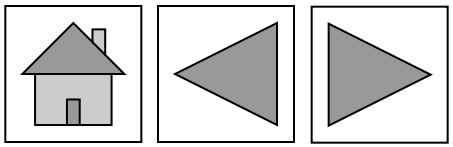


Фреймы (продолжение)

Разрешенные имена

Зарезервированные неявные имена фреймов

- blank – загружает документ в новое окно, не имеющее имени
  <A href="stuff.htm" target="_blank">
- self – загружает документ в текущее окно
  <A href="stuff.htm" target="_self">
- parent – загружает документ в окно родительской фреймовой структуры
  <A href="stuff.htm" target="_parent">
- top – загружает документ в окно фреймовой структуры верхнего уровня по отношению к данному фрейму
  <A href="stuff.htm" target="_top">



Формы

Передача в PHP $_post [‘<name’]
$_get [‘<name’]

<FORM> →
- NAME – имя формы
- Action – указывает URL, по которому передается введенная информация
- Method –
  - POST – в закрытом виде
  - GET – в откр. через строку адреса

Пример:

```
<FORM METHOD=“POST”
  ACTION=“/cgi-bin/comment_script> name = ‘1’
</FORM>
```

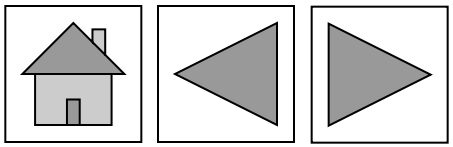
<TEXTAREA> - поле для ввода текста →
- NAME – имя поля
- ROWS – число строк в поле
- COLS – количество символов в строке
- Default text – текст по умолчанию
- Disabled – поле не активное
- readonly- только для чтения

Пример:

```
<FORM>
<TEXTAREA NAME=“coments” ROWS=4 COLS=40>
Default text 1,2,3… - текст по умолчанию
</TEXTAREA>
```

Результат name=value

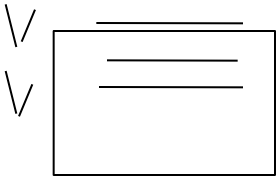
Пример



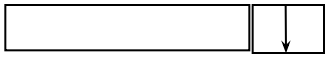
Пример

<SELECT> - выпадающее меню

NAME – имя поля (обязательный)
SIZE – число строк в многострочном меню
MULTIPLE – режим выбора нескольких значений (многостр. меню)



Без MULTIPLE - однострочное меню



```
<FORM>
<SELECT  NAME = “Network”>
<OPTION SELECTED VALUE=“Ethernet”> ETHERNET
<OPTION VALUE = “Token 16”> Token Ring – 16 MB
<OPTION VALUE = “Token 4”> Token Ring – 4 MB
<OPTION VALUE = “Localtalk”> Local Talk
</SELECT>
</FORM>
</BODY>
</HTML>
```

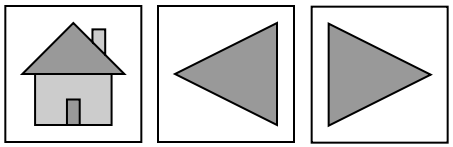
Результат:
name = value
(select) (option)

<OPTION>

VALUE – присваиваемое значение
SELECTED – (по умолчанию)
DISABLED - неактивный

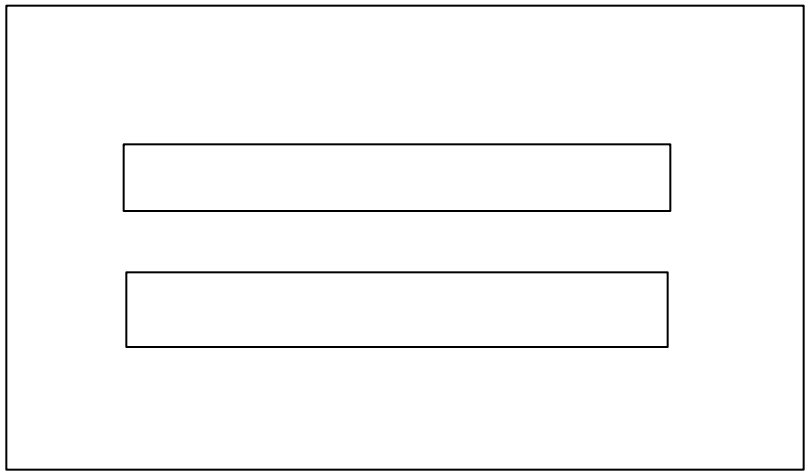
</SELECT>
</FORM>

Назад



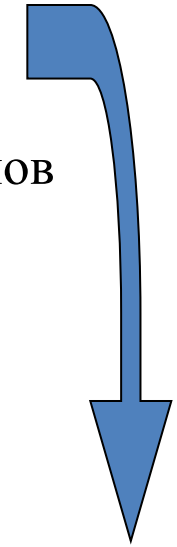
<fieldset> <legend>наименование формы </legend> </fieldset>

Наименование Формы

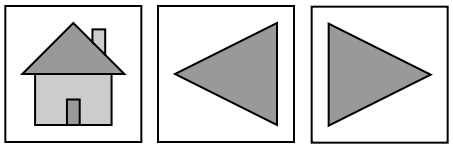


<INPUT>

Атрибуты
NAME - имя поля
SIZE - размер поля
MAXLENGTH - максимальное количество слов
VALUE - значение по умолчанию
CHECKED - флажок/переключатель
TYPE - тип поля



TEXT – ввод строки
PASSWORD – ввод пароля
CHECKBOX – простой флажок
RADIO – сложный флажок
RESET – кнопка «отменить»
SUBMIT – кнопка «переслать»
BUTTON - любое действие
IMAGE - аналог SUBMIT (с индивидуальным рисунком)
FILE - прикрепить файл



TEXT – ввод строки →

```
<FORM>
<INPUT TYPE = “text” name=“phone” value=“текст”>
SIZE=“15” MAXLENGTH=“12”          (по умолчанию)
</FORM>
```

PASSWORD – ввод пароля →

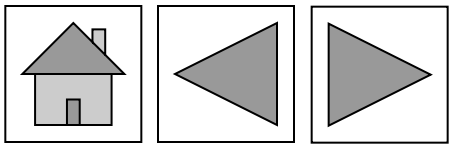
```
<FORM>
Введите пароль <INPUT TYPE = “password”>
 name=“secret_word” SIZE=“30”
 MAXLENGTH=“30”
</FORM>
```

Результат name=on ☑
name=off ☐

CHECKBOX – простой флажок →

```
<FORM>
<INPUT TYPE = “checkbox” name=“checkbox1”>
флажок
<INPUT TYPE = “checkbox” name=“checkbox2”>
CHACKED>установленный флажок
(по умолчанию флажок установленный)
</FORM>
```

☐ - флажок ☑ - установленный флажок

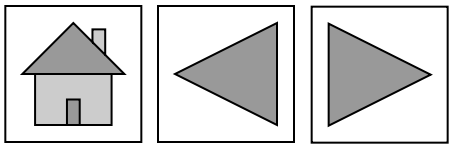


RADIO – сложный флажок

```
<FORM>
<INPUT TYPE = “radio” name=“choice”VALUE=“ch1”>ДА
<INPUT TYPE = “radio” name=“choice”VALUE=“ch2”>Нет
</FORM>
<FORM>
<INPUT TYPE = “radio” name=“choice”VALUE=“ch1”>
CHACKED>Да
<INPUT TYPE = “radio” name=“choice”VALUE=“ch2”>Нет
</FORM>
```

Результат name=value ⊙ Да ○ Нет

Disabled – флажок не активный ⊙ Да ○ Нет



RESET – кнопка «отменить» ⟶

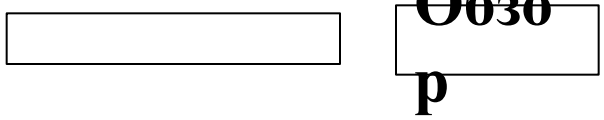
```
<FORM>
<INPUT TYPE = “reset”>
<BR>
<INPUT TYPE=“reset” value=“Отменить!“>
</FORM>
```

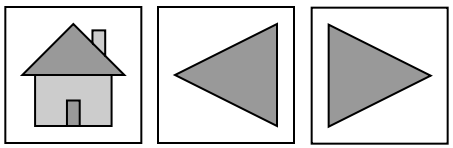
SUBMIT – кнопка «переслать» ⟶

```
<FORM>
<INPUT TYPE = “submit”>
<BR>
<INPUT TYPE=“submit”
value=“Отослать данные“>
</FORM>
```

FILE - прикрепить файл ⟶

```
<FORM>
<INPUT TYPE = “File” name=“file”>
</FORM>
```





Создание навигационных карт ссылок

1. Графиеское изображение карты ссылок

2. Файл определения карты ссылок (форматы CERN NSCA)

Область по умолчанию: default http://www.myserver.com/mypage/index.htm
Прямоугольная область: rect http://www.myserver.com/mypage/rectangle.htm
50, 40, 100, 120
Круглая область: circle http://www.myserver.com/mypage/rectangle.htm
50, 40, 100, 60
Многоугольная область: poly http://www.myserver.com/mypage/rectangle.htm
10, 20 24, 70 84, 45 07, 11 10, 20

3. Программа или сценарий обработки карты ссылок

1. Вызов CGI-программы: <A href = "http://www.myserver.com/cgi-bin/mapfile.map">
<IMG SRC = imagemap.gif ISMAP> </A>

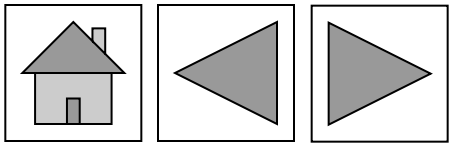
2. Работа с провайдером: <A href = "/cgi-bin/imagemap/~username/image.map">
<IMG SRC = "myimage.gif"> </A>

4. Карты ссылок, Обрабатываемые клиентом

<MAP NAME = "mapname">
<AREA [SHAPE = "shape"] COORDS = "x, y, …" [HREF = "URL"/NOHREF>
</MAP>

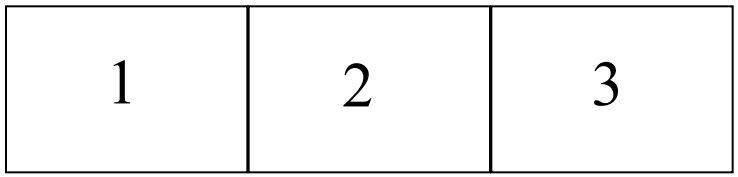
SHAPE – форма области (rect, poly, circle, default)
COORDS – список координат
HREF – URL, на который ссылается область
NOHREF – область – мертвая зона

Пример

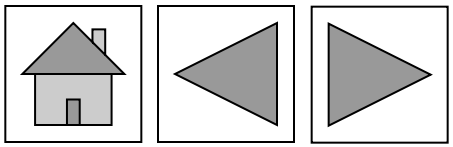


Пример

<MAP NAME = mymap>
<AREA SHAPE = RECT COORDS = “0, 0, 100, 100” HREF = item1.html>
<AREA SHAPE = RECT COORDS = “101, 0, 200, 100” HREF = item2.html>
<AREA SHAPE = RECT COORDS = “201, 0, 300, 100” HREF = item3.html>
</MAP>
<IMG SRC = mymap.gif USEMAP = #mymap> - ссылка на карту ссылок



Назад

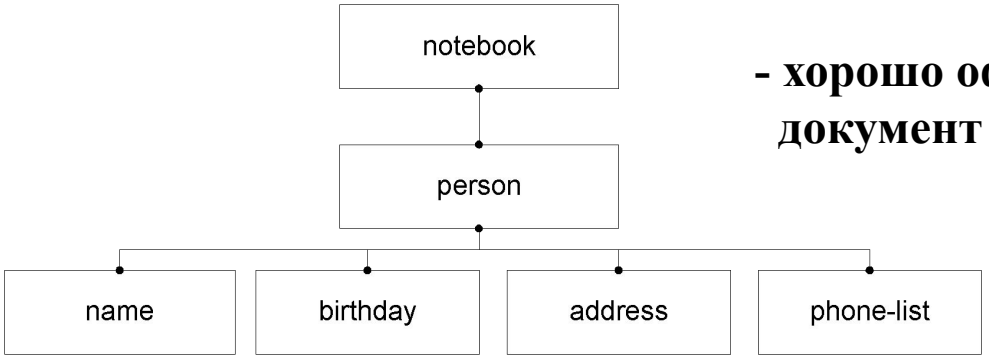


XML

Реализации *XML* →

- *XHTML* – язык *HTML*, приведенный в соответствие с XML
- *MathML* – язык записи математических формул
- *CML* – язык записи химических формул
- *VoxML* – язык записи звуков
- *WML* – язык, применяемый в безпроводной технологии

Верный документ - документ, который соответствует описанию структуры документа или схеме документа (правила написания тегов).



- хорошо оформленный документ

Для описания схемы документа используются язык DTD или XSD, где
DTD - язык описания структуры документа;
XSD - язык описания схемы документа.

(*Extensible Markup Languages* – расширенный язык разметок)

XML (*Extensible Markup Languages* – расширенный язык разметок)

Пролог *XML* →

- <?xml version="1.0"?>
- <?xml version="1.0" ecoding="UTF-8"?>
- <?xml version="1.0" ecoding="windows-1251" standalone="yes"?>

  yes – по умолчанию;

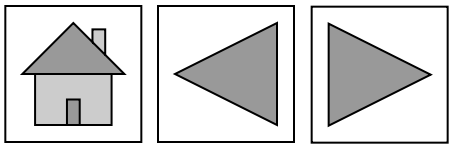
  no – ссылка на внешний документ

Корневой элемент → (root)

- <! *DOCTYPE notebook [описание DTD]*> - описание внутри документа
- <! *DOCTYPE notebook* (имя root) [*SYSTEM* / *PUBLIC*] (если определение документа записывается в отдельном файле) *"ntb.dtd"*>

Элементы *XML* →

- <*surname*> Сидорова </*surname*>
- <*br* /> для пустых элементов



Комментарий[<!--Коментарий-->

Атрибуты → <name first="Иван" second="Петрович" surname="Сидоров" /> значение атрибутов ""

<city type="город" >Москва </ city>

посёлок
деревня

Секция *CDATA* →

<[CDATA[Сидорова <surname>]] <surname>

↓

<progcode>

<![CDATA [if (X < MAX_VALUE && x > MIN_VALUE) proc1(x);]]>

</progcode>

не допускается вложенность *CDATA*

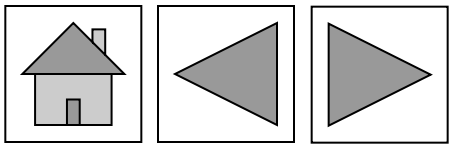
Внутри секции *CDATA*-набор символов, который необходимо передать.

Пространство имен → *XML*

<ntb:notebook xmlns:ntb="http://...ntbml">

<ntb:city type="посёлок" Горелово="/" > </ntb:city>

(префикс)



Таблицы

→

SUPPLIER

SNUM	SNAME	LOC
10123	Иванов	Санкт-Петербург
10212	Петров	Львов

PRODUCT

PNUM	PNAME	Color	WEIGHT
012	Болт	Синий	25
013	Гайка	Черный	20
…	…	…	…

SP

SNAME	PNAME	QTY
10123	гайка	500
10123	болт	800
10212	болт	300

SQL Поставка

```
<? xml version = "1.0"
      encoding = "Windows - 1251"?>
<? DOCTYPE supplprod
      SYSTEM "suppl-prod.dtd">
< sp : supplprod xmlns : sp = "http://..." >
< sp: suppl_prod
>
< sp : snum  10123
>
< sp : sname > Иванов </ sp :
> sname
< sp : loc  Санкт-Петербург / sp:loc
>
< sp : pname > гайка </sp :
> pname
< sp : qty > 500 </sp : qty
> / sp : suppl_prod
> sp : suppl_prod
>
< / sp : suppl_prod >
```

- корневой элемент – имя БД
- имя элемента – имя таблицы
- имя столбца – значение строки

Язык DTD *(Document type Definition)*

Обьявление типа элементов → *< !ELEMENT*

< !ELEMENT br EMPTY/ > – пустой элемент *< br >*

< !ELEMENT something ANY > – любой элемент

< !ELEMENT name (#PCDATA) – текстовый элемент (без вложений)

< !ELEMENT prod (sp : snum, sp : sname, sp : loc...) > – элементы с вложением

↓

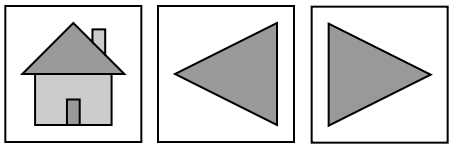
< !ELEMENT sp : supplprod(sp : suppl _ prod) >*

< !ELEMENT bk : chapter(bk : title,(bk : section, bk : listing, bk : picture*)*, bk : conclusion?) >*

? элемент или список может встретится оди н или ноль раз

* элемент или список мож ет встретится нуль или несколько раз

+ – элемент или список может встретися один или несколько раз



Обьявление атрибутов → *< !ATTLIST*

Типы атрибутов

CDATA – строка символов

ID – уникальный идентификатор

IDREF – идентификатор-ссылка на другие элементы

ENTITY – имя неповторяемой анализатором сущности

<!*ELEMENT name* (#*PCDATA*) >
<!*ATTLIST name reg _num ID #REQUIRED* > ⇒
<!*ELEMENT post* (#*PCDATA*) >
<!*ATTLIST post ref IDREF #IMPLIED* >

< name reg_num=“1045”>
Иванов </ name>
< name reg_num=“1052”
Петров < / name>
<<rost ref = “ 1045”>
Зав. отделом </ post>

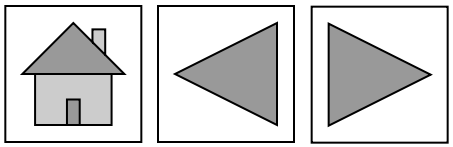
Иванов = зав. отделом

Признак обязательности:

#*REQUIRED* – обязательно записывать в элементе

#*IMPLIED* – необязателен, у него нет значения по умолчанию

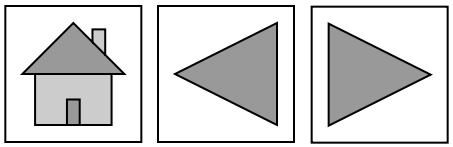
#*FIXED* – у атрибута только одно значение



Примеры

```
<!ATTLIST name
first CDATA #REQUIRED
second CDATA#IMPLIED
surname CDATA#REQUIRED>

<!ATTLIST bk:listing
bk:number ID #REQUIRED
bk:head CDATA #REQUIRED>
```



обьявление сущности (*entity*) →

Внутренние сущности: - задаются при объявлении сущности

<!ENTITY author "

ссылка на сущность

<!ENTITY &cover "; Название &author ;" title. >

Внешние сущности: -содержаться в отдельных файлах

<!ENTITY tel

SYSTEM "http://www.tty.com/TelDef.xml">

<!ENTITY tel

PUBLIC "_//DTD/Tele..." "http://www.tty.com/TelDef.xml">

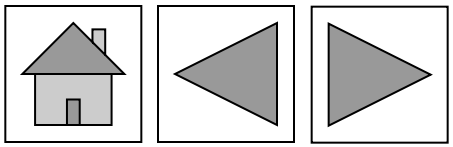
PUBLIC – общеизвестное обьявление

Параметризованная сущность:- используеться только внутри объявления DTD

<!ENTITY % lang "ru_Ru">

%lang; – ссылка на параметризированную сущность

Используется только внутр и описания *DTD*



не разбираемая программой анализатором →

Сущность которая не обрабатывается средствами *XML*

<!ENTITY tel _logo

SYSTEM "/ image / TelDef logo 32x16.gif" NDATA image – gif >

NDATA – определяет не разбираемую сущность

image – gif – программа, которая открывает изображение

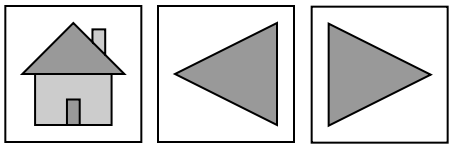
Обьявление обозначения (NOTATION) →

<!NOTATION image_gif SYSTEM "viewer.exe"

связывает image – gif с программой обработки изображений в файле viewer.exe

Пример описания *DTD*. Записная книжка

```
<!ELEMENT notebook (person)* >
<!ELEMENT person (name, birthday?,
  address*, phone – list?) >
<!ELEMENT name EMPTY >
<!ATTLIST name
    first CDATA# IMPLIED
    second CDATA# IMPLIED
    surname CDATA# REQUIRED >
<!ELEMENT birthday (# PCDATA)
<!ELEMENT address (street , city, zip)? >
<!ELEMENT street (# PCDATA) >
<!ELEMENT city (# PCDATA) >
<!ATTLIST city
        type (город поселок деревня город)"  ">
<!ELEMENT zip (# PCDATA) >
<!ELEMENT phone – list (work – phone*,
                        home - phone*) >
<!ELEMENT work – phone (# PCDATA) >
<!ELEMENT home - phone (# PCDATA) >
```



Язык *XSD* (Описание схемы документа – *XML Shema Definition*)

XSD – реализация XML. Корневой элемент shema.

Простые типы *XSD* →

- *float*, *double* – вещественные числа
- *integer* – целые числа
- *string* – строка символов
- *date Time* – дата и время
- *any URL* – адреса URL
- *zip* – почтовый индекс
- *Qname* – расширеное имя тэга или атрибута

→

Тип:

- $time \rightarrow hh:mm:ss$
- $date \rightarrow CCYY-MM-DD$
- $gYearMonth \rightarrow CCYY-MM$
- $gMonthYear \rightarrow -MM-DD$
- $gYear \rightarrow CCYY$
- $gMonth \rightarrow -MM-$
- $gDay \rightarrow -DD-$

Пример: < *xsd* : *simpleType* определение типа ">

< / *xsd* : *simpleType* >

Сужение

restriction →

```
< xsd : simpleType name = "zip" >
< xsd : restriction base = "xsd : string" >
< xsd : pattern value = "[0 – 9]{6}" / >
```

6 арабских цифр

фасетка “регулярное выражение“

pattern – почтовый индекс zip

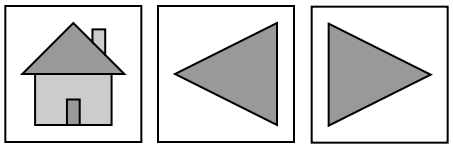
6 арабских цифр

```
< /xsd : restriction >
< xsd : simpleType >
< xsd : simpleTypename = "zip" >
< xsd : restriction base = "xsd : positive Integer" >
< xsd : minInclusive value = "100000" / >
< xsd : max Inclusive value = "999999" / >
< /xsd : restriction >
< /xsd : simpleType >
```

→

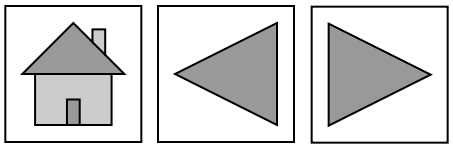
$100000 < zip < 999999$

Фасетка (facets) “наибольшее значение“



facets =

- *maxExclusive* - наибольшее значение, которое уже не входит в определяемый тип.
- *maxInclusive* - наибольшее значение определяемого типа
- *minExclusive* - наименьшее значение, которое уже не входящее в определяемый тип
- *minInclusive* - наименьшее значение определяемого типа
- *totalDigits* - общее количество цифр в числовом типе
- *fractionDigits* количество цифр в дробной части числа
- *length* - длина значений определяемого типа
- *maxlength* - найбольша длина значений определяемоготипа
- *minlength* - наименьшая длина значений определяемого типа
- *pattern* - регулярное выражение



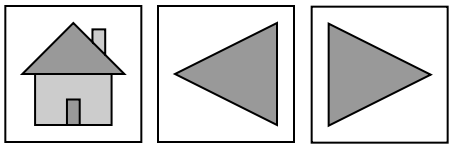
fundamental facets

Базисные фасетки

$=$

- *ordered* →
  - *false* – тип не упорядочен
  - *partial* – тип частично упорядочен
  - *total* – тип полностью упорядочен
- *bounded* →
  - *true* – тип ограничен значением
  - *false* – тип не ограничен значением
- *numeric* →
  - *true* – тип числовой
  - *false* – тип не числовой

Используются как атрибуты тэгов в фасетках



Список *list* →

$< days > 21\ \ 34\ \ 55\ \ 46 < /days >$

↓ схема

$< xsd : simpleType\ name\ <\ xsd : element\ name\ =\ \text{“}days\text{”}$
$type = \text{“}list\ of\ Integer\text{”}\ /> \rightarrow$ список из целых чисел < 5

$< xsd : restriction >$

$< xsd : simple\ Type >$

$< xsd : list\ item\ Type = "xsd : Integer"\ />$

$< xsd : Simple\ Type >$ → тип элементов

$< xsd : \max\ Length\ value = "5"\ />$

$< /xsd : restriction >$

$< xsd : Simple\ Type >$

```
xsd:union memberType="xsd:string
xsd:integer   Listofinteger“   />
<xsd: attribute name = “size” >
<xsd:simpleType>  {имена встроенных типов }
<xsd:union>
<xsd:simpleType>
<xsd:restriction base="xsd:positive Integer">
<xsd:min Inclusive value ="8 />
<xsd:max Inclusive value"72 />
</xsd:restriction>
</xsd:SimpleType>
+<xsd:SimpleType>
<xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="smale"/>
<xsd:enumeration value="medium"/>
<xsd:enumeration value="large"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:union>
</xsd:Simple type>
</xsd:attribute>
```

Обьединение *union* →

→

```
<font size="large">
Глава 1 </font>

<font size="12">
Простой текст </font>
```

Объявление элементов *element* →

```
< xsd : element name = " имя элемента "  type = " тип элемента "
minOccurs = " минимальное число появлений "
maxOccurs = " максимальное число появлений элемента в документе " />
< xsd : element name = " degree " type = " xsd : nonPositiveInteger " />
```

по умолчанию = 1

Обьявление атрибутов *attribute* →

```
< xsd : attribute name = " имя атрибута "  type = " тип атрибута "
base = " базовый тип "  use = " обязательность атрибута "  default = " значение по умолчанию " />
```

use =
- *optional* – необязателен
- *required* – обязателен
- *prohibited* – неприменим (определение подтипа с целью отменить атрибуты базового типа)

↓

```
< xsd : attribute name = " id " type = " positive Integer "
                                use = " required " / >
< xsd : attribute name = " name " type = " NCName "
                                default = " anonymous "
                                / >
```

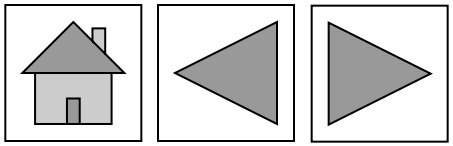
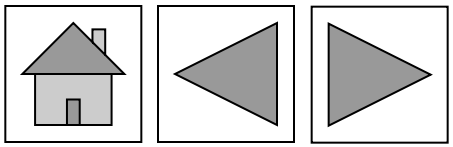


Схема *XSD* книжки

```
< xsd : shema xmlus : xsd = " http : ...XML Schema" >
< xsd : element name = " notebook" type = " notebook Type"  /
                                                            >
< xsd : complex Type name = " notebook Type" >
            < xsd : element name = " person" type = " person Type"
               min Occurs = "0" max Occurs = "unbounded"  /
                                                          >
            < / xsd : complex Type >
< xsd : complex Type name = " person  Type" >
< xsd : sequence >
      < xsd : element name = " name" >
         < xsd : complex  Type"
            < xsd : attribute  name = " first" type = " xsd : string"
               use = "optional"  /
                                 >
            < xsd : attribute name = " sec ond" type = " xsd : string"
               use = "optional"  /
                                 >
                 
            < / xsd : complex Type >
         < / xsd : complex Type >
      < xsd : element >
                 
< / xsd : sequence >
< / xsd : complex Type >
< / xsd : shema >
```



Язык *XLINK*

Пространство имен *XLINK* → < *someElement xmlns : xlink =* *“http://www.w3.org/1999/xlink”* > Содержание элемента < */someElement* >

Атрибуты *XLINK* →

- *type* – тип ссылки → *simple* – простая ссылка; *extended* – расширеная ссылка; *resource* – информационный ресурс; *locator* – указатель на ресурс; *arc* – дуга графа; *title* – заголовок; *none* – остальные элементы не имеют отношения к ссылкам
- *href* – адрес ресурса
- *show* – способ показа ресурса
- *actuate* – момент активации ссылки
- *label,from,to* – начальные и конечные пункты ссылок
- *role, arcrole, title* – смысл ссылки

Атрибут *Simple* → простая ссылка

Simple – аналог *a, img*

```
< some Lnk xmLns : xlink = " http :..."
    xlink : type = " simple"
    xlink : title = " Здесь хранится классификатор "
    xlink : href = " http :...k1d2012.xml" >
< / some Lnk >
```

Атрибут *Extended* → расширенная ссылка

Extended – содержит внутри себя другие ссылки

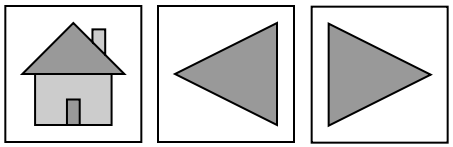
```
< mult link  xmlns : xlink = " http :..."
    xlink : type = " extended"
    xlink : title = " Связь документов "  >
< src  xlink : type = " resource" xlink : label = " s0012"  / >
< tgt  xlink : type = " locator"
        xlink : href = " http :...udr01.xml"
        xlink : label = " f0012"/ >
< ref  xlink : type = " arc" from = "s0012" to "f0012"  / >
< / mult link >
```

информ. ресурс

указатель на ресурс

элемент

ссылка



Атрибут

RESOURCE → [Отмечает локальный ресурс, в котором записана ссылка

Атрибут

LOCATOR →

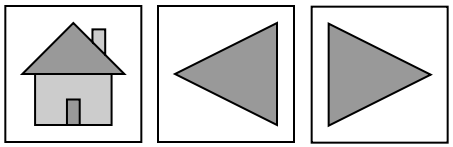
- Описывает удаленный ресурс
- Используется при создании ссылки дуги типа *arc*

$$< my\ Biogr\ \ xlink:type = \underbrace{"locator"}_{\text{указатель на ресурс}}$$

$$xlink:href = "http:....xml"$$

$$xlink:label = "L1234"/>$$

- Ссылка типа *locator* применяется только внутри *extended*



Атрибут *ARC* →

Описание связи между двумя ресурсами

From, *to* используется в дугах указания начальной и конечной точек

```
<tplinl xlink:type="extended">
  <scresource xlink:type="locator" xlink:label="loc"
      xlink:href="http://...reso24.xml"/>
  <scresource xlink:type="locator" xlink:label="base"
      xlink:href="http://...reso43.xml"/>
  <load xlink:type="arc" xlink:from="loc"
      xlink:to="base"/>
</tplink>
```

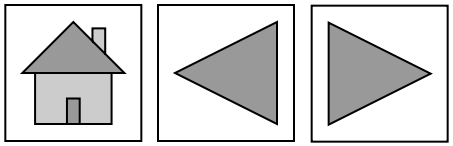
указ. на ресурс удалённый

указ. на ресурс удалённый

Атрибут *TITLE* →

Дает описание расширенной ссылке

```
<rem xlink:type="title">
        Другие материалы по теме
</rem>
```



Атрибут *Show* → Используется в ссылках типа *simple* и *arc* для отображения на экране

Значение атрибута *Show* :

new – показать в новом месте

replace – показать в месте где была ссылка на ресурс

embed – при показе первоначальный ресурс не подвергается преобразованию

other – способ представления описания в других елементах документа

none – способ представления не описывается ни водном элементе документа

```
< sing  xlink : type = "simple"
xlink : href  = "http : / / ...sigma.gif " >
xlink : show = "embed"
xlink : title = "Знак суммы" / >
```

Атрибут *actuate* →

Используется в ссылках *simple* и *arc* для определения момента времени активации ссылки:

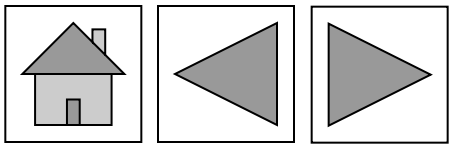
on load – при загрузке документа

on Request – по событию

other – способ активации описан в других элементах

none – способ активации не описан ни в одном элементе документа

```
< sing   xlink : type = " simple "
     xlink : href  = " http : / / ...intro.xml "
     xlink : actuate = " on load "  / >
```



Атрибут *role* →

Не используется в типах: *arc*

Атрибут *role* указывает на ресурс, описывающийссылку

$< problem\ \ xlink:type = "simple"$

$xlink:href = "..."$ - адрес ссылки

$xlink:role = "..."$ - адрес ресурса описания ссылки

Решение интегрального уравнения

$</ problem >$

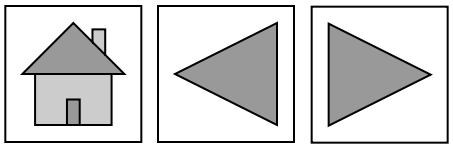
Атрибут *arcrole* →

Аналогично *role* применяется в ссылках , *arc*

$< load\ \ xlink:type = "arc"$

$xlink:\overset{arcrol}{\underset{e}{}} = "..."$

$xlink:from = "cont"\ xlink:to = "base"\ xlink:actuate = "onload"\ />$



Уточненные ссылки

XPointer

Простые указатели →

```
<myLink xlink:type="simple"
        xlink:href="myDocument#label02" />
```

– ссылка с внешнего ресурса

```
<myLink xlink:type="simple"
        xlink:href="#label02" />
```

– ссылка внутри документа

Указатели, основанные на схеме →

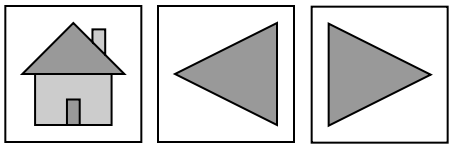
```
xpointer(/book/chapter/section) element(color/3)
```

↓

```
<book>
    <chapter id="ch5" name="ch5" title="
        <section name="sect1">
            ...
        </section>
    </chapter>
</book>
```

↓

```
<attr id = “color”
<rgb…          </rgb>
<hsb…          />
<cmyk c=“25”…/>
<icc…          />
</attr>
```



Использование указателей в ссылках →

```
<myLink xlink:type="simple"
        xlink:href="mydoc.xml#element(color/3)"/>
```

Схема *element* →

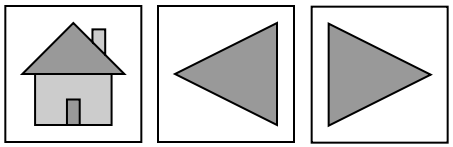
element(11/3/2)

element(*sect*3*a*/2)

element(*sect*3*a*)

→

```
<contact numb="5">
 <section id="sect1a">
   <paragraph>…</paragraph>
        ☒
 </section>
 <section id="sect2a">
        ☒
 </section>
 <section id="sect3a">
   <paragraph>…</paragraph>
 </section>
```



Схема

xpointer ( ) →

- *xpointer* (/*contract*)
- *xpointer* (/*contract* / *section*)
- *xponter* (/*contract* / *section* / *paragraph*)
- *xpointer* (/*contract* / *section* / *)
- *xpointer* (/*contract* / * / *paragraph*)
- *xpointer* (/ / *)
- *xpointer* (/*contract* / *section*[1])
- *xpointer* (/*contract* / *section* [*connt*(*) = 2])
- *xpointer* (/*contract* / *section*[2] / *paragraph*[1])

* все

. текущий элемент

.. родительский элемент

[n] порядковый номер из выбраных элементов

< /*contract* >

< *contract numb* = "5" >

< *section id* = ... >

< *paragraph* > ... < /*paragraph* >

[redacted]

< /*section* >

< *remark* > ... < /*remark* >

< *section id* = "*sect* 2*a*" >

[redacted]

< /*section* >

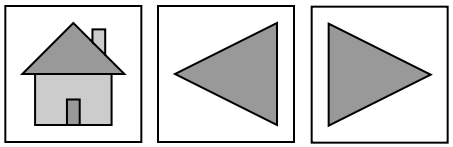
< *remark* > ... < /*remark* >

< *section id* = "*sect* 3*a*" ... >

[redacted]

< /*section* >

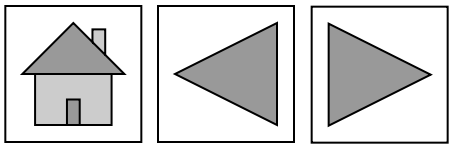
< /*contract* >



Схема

$xmlns(\ ) \rightarrow$

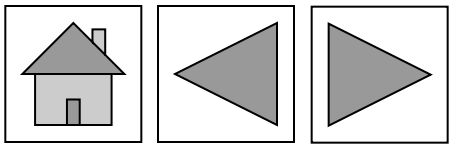
$$\left[\begin{array}{l}
xmlns\ (abc = http://example.com/ns/abc) \\
1)\ xmlns\,(img = http://example.org/image) \\
\qquad\qquad img : rect\,(10,10,50,50) \\
2) < customer\ \ xmlns = "http://example.org/customer" > \\
\qquad\qquad < name\ \ xmlns = http://example.org/\ personal-info" > \\
\quad < customer > \qquad\qquad\qquad\qquad John\ Doe < /name > \\
xmlns\ \ (c = http://example.org.customer) \\
xmlns\ \ (p = http://example.org/\ \ personal-info) \\
xpointer\,(/c : customer/\ p : name)
\end{array}\right.$$



Язык *XPath*

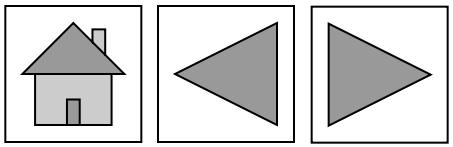
Узлы дерева документа →

- узлы документы (*document nodes*)
- узлы элементы (*element nodes*)
- узлы атрибуты (*attribute nodes*)
- узлы пространства имен (*namespace nodes*)
- узлы инструкций по обработке (*proccesing instruction nodes*)
- узлы коммента рии (*comment nodes*)
- текстовые узлы (*text nodes*)

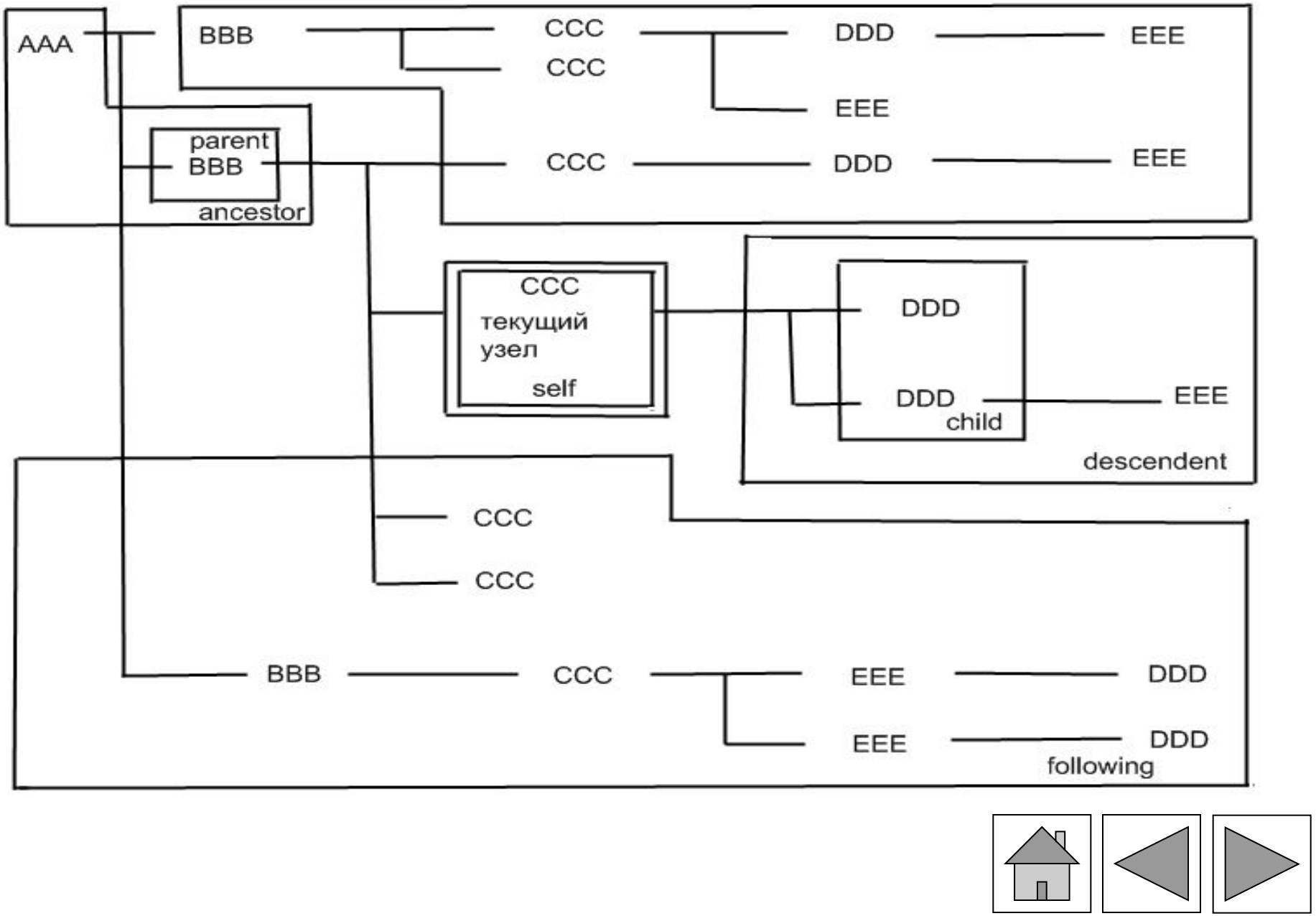


Оси поиска →

- *ось* :: *текст узла*[*предикат*] → *child* :: *section*[1] → *section*[1]
-
  - *self* – текущий узел
  - *child* – узлы потомки, кроме узлов-атрибутов и узлов пространств имен
  - *descendant* – узлы потомки с их потомками
  - *descendant – or – self* – обьединение *descendant* и *selt*
  - *following* – все узлы, лежащие "ниже" текущего узла
  - *ancestor* – все узлы-предки текущего узла
  - *preceding* – узлы, предшествующие текущему узлу
  - *parent* – непосредственный предок текущего узла
- По умолчанию *child*

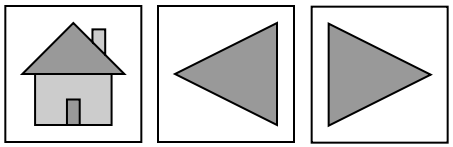


Области, определяемые осями



Тест по имени узла *name test* →

- *self* текущий узел дерева
- *child* непосредственные потомки узла *person*
- *descendant* все потомки узла *person*
- *descendant – or – self* все потомки узла *person* и сам узел *person*
- *parent* родительский узел узла *person*
- *ancestor* все предки узла *person*
- *attribute* все атрибуты узла *person*

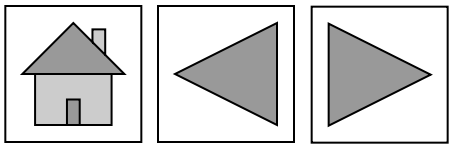


Тест по виду узла → (*kind test*)

- *node* () – отбирает узел любого вида
- *text* () – отбирает текстовые узлы
- *element* () – отбирает все узлы элементы
- *element* (*name*) – отбирает узлы элементы с именем *name*
- *element* (*name*, *type*) – отбирает узлы элементы с именем *name* и типом *type*
- *attribute* () – отбирает все узлы атрибуты
- *attribute* (*name*) – отбирает узлы атрибуты с именем *name*
- *attribute* (*name*, *type*) – отбирает узлы атрибуты с именем *name* и типом *type*
- *document node* () – отбирает все корневые узлы документа

Пример

- *child* :: *element* () – отбирает узлы потомки вида узлы-элементы
- *self* :: *comment* () – отбирает текущий узел-комментарий

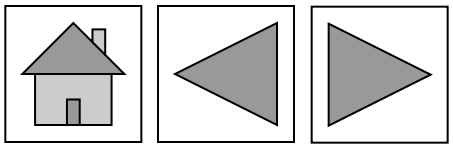


Предикаты →

- /*/CCC/BBB[1] – первый из элементов BBB, вложенный в CCC, вложенный в корневой элемент документа
- /*/CCC/BBB[last()] – последний из элементов BBB, вложенный в CCC, вложенный в корневой элемент документа
- /*[count(CCC)=2] – все элементы документа с двумя вложенными элементами CCC
- /*[count(*)=2] – все элементы документа с двумя любыми вложенными документами
- //DDD[@name] – все элементы документа, имеющие атрибут DDD с атрибутом name
- //DDD[@id="3"] – все элементы документа, имеющие атрибут DDD с атрибутом id=3
- //DDD[not(@name)] – все элементы документа, не имеющие атрибут DDD с атрибутом name

if →

- *if* (выражение1) *then*(выражение2) *else*(выражение3)
- *if* ($plsex="M") then" father" else" mother"
- *if* (/count/price=0) then/ count / price else() – пустое выражение
- Ветвь *else* – обязательна!!!



Циклы → for $n in выражение1 return выражение2

for $n in //BBB return $n/CCC – выбрать все узлы *CCC*, вложенные в узлы *BBB*

for $X in //BBB return (for $y in $x/CCC return $y/DDD) – вложенный цикл

Кванторы (*quantifiers*) существования → some $n in выражение1 satisfies выражение2

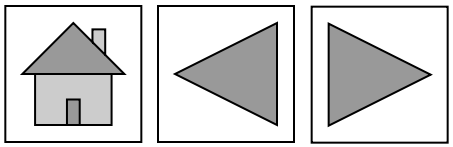
some $n in //person?name satisfies $n = "Федор" – существует хотя бы один сотрудник по имени "Федор"

Квантор всеобщности → every $n in выражение1 satisfies выражение2

= *true*, если выражение2 всегда равно *true*

= *false*, если выражение2 один раз принимает значение *false*

every $n in //person/education satiesfies $n = "ee" – все ли сотрудники имеют высшее образование



Операции с множествами →

- *union* () выбирает все элементы и , вложенные во все элементы *AAA*

doc ("*bids.xml*") / * / *bid* [*bid _ date* > *date* ("2002 – 01 – 01")]

выделяет узлы продукции стоимостью > 1000 идатой> 2002-01-01

- выбирает из первой последовательности

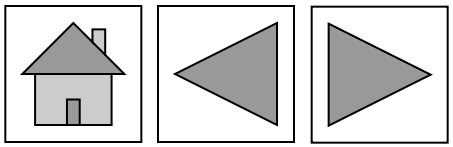
except – дополнение узлы,не содержащиеся в о второй последовательности

Язык запросов *XQuery*

Прямой конструктор элемента →

```
< person id = "92 – 3456" sex = "male" >
    < name > Иван Петрович < / name >
    < age > 30 < / age >
< person >
```

Выражение в содержимом конструктора → [< *age* > {10 + 20} < / *age* >]



Выражение в атрибутах конструктора →

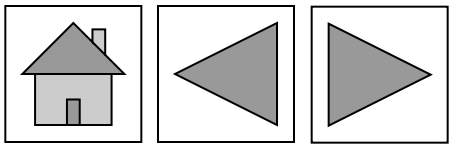
$$\begin{bmatrix} < person\ id = "92-3456" sex = "\{/notion/sex[1]\}" > \\ < person\ id = 9\{1+1\}-3456" sex = "male" > \end{bmatrix}$$

Вычисляемые конструкторы →

$$\begin{bmatrix} element\ \ person\{ \\ \quad attribute\ id\{"92-3456'\} \\ \quad attribute\ sex\{/notion/sex[1]\} \\ \quad Имя\ Петров\{" \qquad "\} \\ \quad element\ age\{10+20\} \\ \} \end{bmatrix}$$

$\Rightarrow$

element
{/notion/name[1]}
{attribute id{"92-3456"}
attribute
{/notion/attr[1]}
{/notion/sex[1]}
Имя Петров {" "}
element age {10+20}
}



Выражение запроса *FLWOR* →

```
let $n := ("Иванов","Петр")
return <name> {$n} </name>
```
⇒ <name> Петр Иванов </ name>

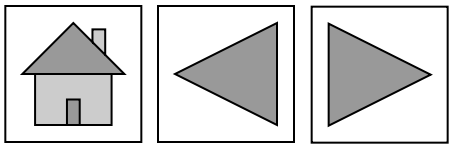
```
for $n in ("Иванов","Петр")
return <name>{$n}</name>
```
⇒ <name> Петр Иванов </ name>
<name> Иванов </ name>

```
for $n at $i in ("Иванов","Петр")
```
⇒ ($n= “Петр”, $i=1)
($n= “Иванов”, $i=2)

```
for $i in (10,20), $j in (1 to 3)
return <x> {$i, ",", $j)} </x>
```
⇒ <x> 10,1 </ x>
<x> 10,2 </ x>
<x> 10,3 </ x>
<x> 20,1 </ x>
<x> 20,2 </ x>
<x> 20,3 </ x>

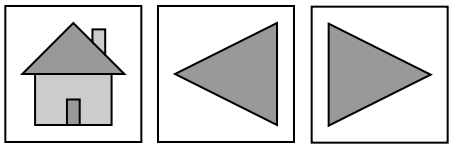
```
for $x in /count/incr
where $x > 0
return $x
```
Если выражение истина, то
Выполняется выражение *return*

```
for $p in ("Иванов","Петр")
order by $n descending
return <name> {$p} </name>
```
order by - сортировка
- *ascending – по возрастанию*
- *descending – по убыванию*

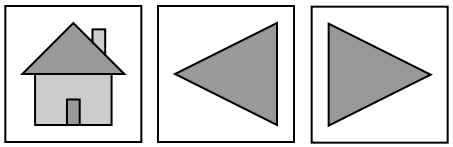


Файл *bib.xml*

```
<bib>
  <book year="2003">
    <title>Протоколы TCP/IP</title>
    <author>
      <last>Стивенс</last>
      <first>first</first>
    </author>
    <publisher>Издательский Дом Диалектика</publisher>
    <price>220.00</price>
  </book>
  <book year="1992">
    <title>Advanced Programming in the...</title>
    <author>
      <last>Stevens</last>
      <first>W.</first>
    </author>
    <publisher>Addison-Wesley...</publisher>
    <price>65.95</price>
  </book>
```



```
<book year="2000">
    <title>Data on the Web...</title>
    <author>
        <last>Abiteboul</last>
        <first>Serge</first>
    </author>
    <author>
        <last>Buneman</last>
        <first>Peter</first>
    </author>
    <author>
        <last>Sucin</last>
        <first>Dan</first>
    </author>
    <publisher>Morgan Kaufmann...</publisher>
    <price>39.95</price>
</book>
<book year="1999">
    
</book>
</bib>
```



Выражение запроса *FLWOR* →

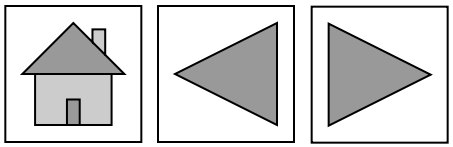
[for < $a1 > at < $i1 > in < Выражение 1 >,
  < $a2 > at < $i2 > in < Выражение 2 >,...
let < b1 > as < Выражение 3 >,
< $b2 > as < Выражение 4 >,...
[where Выражение 5 >]
[order by Выражение 6 >]
return Выражение 7 >

Пример 1: Выделить из списка книг название и год издания книг, опубликованых *Addison_Wesley* после 1991 года

```
<bib>{
  for $b in doc("bib.xml")/bib/book
  where $b/publisher="Addison_Wesley" and $b/@year>1991
  return
      <book year="{$b/@year}">
            {$b/title}
      </book>
    }
</bib>
```

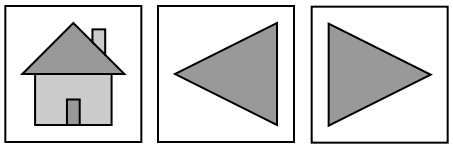
⇒

```
< bib >
   < book year="1992">
        < title > Advanced… </ title >
   </ book>
</bib>
```



Пример 2: Получить список названий книг и их авторов

```
<results>{
    for $b in doc("bib.xml")/bib/book,
        $t in $b/title,
        $a in $b/author
    return
        <result>
            {$t}
            {$a}
        </result>
}
</result>
```



Пример 3: Выбрать книги название которых заканчивается на "02"
и узлы, содержащие слово "*suciu*".

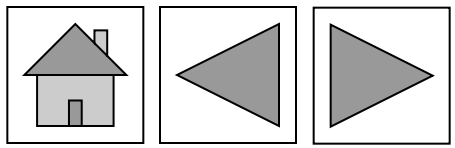
```
for $b in doc("bib.xml") / /book
let $l := $b / *[contains (string(.),"suciu")
and nds – with(local – name(.),"02")]
where exists ($l)
return
    < book >
      {$b / title}
      {$l}
    < /book >
```

⇒

```
< book >
   < title > Data on the Web </ title >
   < author >
      < last > Suciu </ last >
      < first > Dan  </ first >
   </ author >
</ book >
```

Оператор варианта →

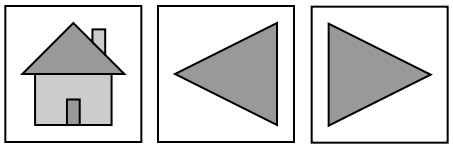
```
typeswitch (/ / address)
    case element (*, USAddress) return / /address / state
    case element(*, RussiaAddress) return / /address / region
    default return "Unknow address type"
```



Пролог → [*xquery version* "1.0"

declare namespace xyz = "*http* : / / *some.domain* / *myns*"

определение пространства имен



Каскадные стили (CSS)
Cascading style sheets

Определение *CSS* →

```
< Мета http _ equiv = "Content _ Style _ Type"
  content = "text / CSS" >
```

Компоновка *CSS* →
файл. *CSS*

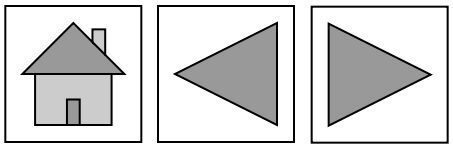
связь файла *CSS* с *HTML* документа

```
< LINK  REL = STYLESSHET  HREF =
"http : / / www...mysheet.CSS" TYPE = "text.CSS" >
```

(описание стилей находится в отдельном ф айле)

Внедрение *CSS*
описание *CSS* внутри
файла →

```
< / HEAD >
< STYLE  TYPE = "text / CSS" >
      Здесь находятся определения стиля
< STYLE >
      (пределения стилей работают только внут ри файла)
< BODY >
```



Встроенный *CSS* описание *CSS* → внутри тэга

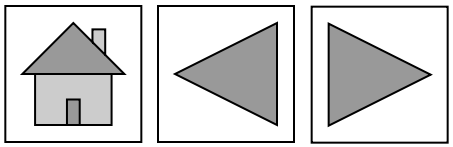
```
<H1 STYLE = "color : blue" >      ...    </H1>
<DIV  STYLE = "color : blue" >
<H1 > Заголовок < /H1 >
<P > Абзац будет выведен браузером
        синим цветом < /P >
< /DIV >
```

```
< SPAN  STYLE = "color: blue" >
< /SPAN >
```

текстовый блок

(встроенный *CSS* имеет приоритет над внедренным или связанным)

SPAN задает стиль для нескольких символов



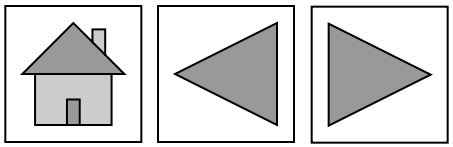
Обьединение листов → стилей

```
< LINK TYPE = "text / CSS" REL = "alternate stylesheet"
TITLE = "Example" HREF = first.CSS >
< LINK TYPE = "text/CSS" REL = "alternate stylesheet"
TITLE = "Example" HREF = second.CSS >
```

Конкурирующие *CSS* →

1. *CSS* автора > *CSS* пользователя > *CSS* браузера
2. Оперативные *CSS* > внедренные *CSS* > связаные *CSS*
3. *H1 (color : red !important font – weight : bolt*
   font – family : sans – serif !important)

   (*important*) → max
4. Классы > переопределение свойств тэга



Создание *CSS* для разных типов представлений →

< style type = "text / CSS" media = "screen" >

Определение стилей

< /style >

media →
- *screen* – дисплей
- *print* – принтер
- ⋮
- *all* – все устройства

< STYLE TYPE = "text / CSS" media = " print" >

Определение стилей

< /STYLE >

Правила *CSS* →

H1 {color : blue}

P { font – size : 1 pt}

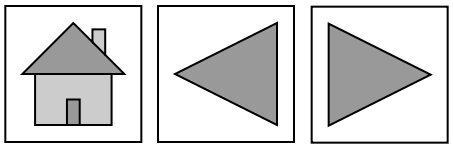
H1 {color : blue, font – size : 2 pt, text – align : center}

P, UL, LI { font – size : 2 pt}

Strong {font-weight:normal}

pt – пункты (1..7) 1pt - 8px
2pt - 10px
↓
7pt – 36px

не будет видна жирность, но будет индексация



Наследование *CSS* (контекстные селекторы) →

- *OL LI (list – style – type : square)* – из нумерованного в маркерованный (для *LI* в списке *OL*)
- *UL LI (list – style – type : decimal)* – из маркерованного в нумерованный (для *LI* в списке *UL*)
- *p span {color : blue}* - переопределение свойств тэга span внутри параграфов.

Классы *CSS* →

- *H1.blue {color : blue}*
- *H1.red {color : red}*
- *H1.black {color : black}*

определение класса для дескриптора

< H1 CLASS = red > Красный заголовок *< /H1 >*

.red {color : red} определение класса для любого дескриптора

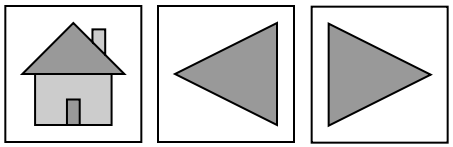
Ul li.forlist {background : blue} - изменение свойств li для маркерованного списка

Специфический элемент *CSS* →

< P ID = test > текстовый абзац *< /P >*

#text {color : red} определение класса для данного элемента

На 1 странице не может быть 2 одинаковых ID. Определяются глобальные классы



Псевдоклассы *CSS* →

Псевдокласс -специальный селектор, определяющий вид *HTML* в определенный момент

селектор:псевдокласс - {свойство:значение}

A link {color : blue} – использованные гиперссылки синие

A active {color : red} – активные гиперссылки красные

A visited {color : yellow} – посещенные гиперссылки желтые

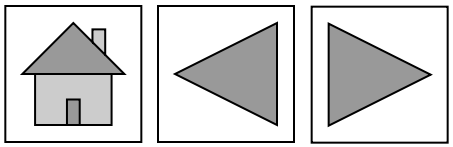
A: hover {color : red} – при наведении мышки

P: first – line {font – weight : bold} – первая строка жирная

P: first – letter {font – weight : bold} – первая буква жирная

Скрытие *CSS* от старых браузеров →

```
< style  type = "text / CSS" >
<! – –
H1 {color : red}
– – >
< / style >
```



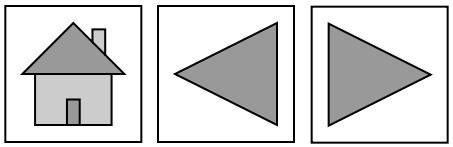
PHP

Интеграция PHP и HTML

Переход в PHP

1. <php - стандартные тэги
2. <? – короткие тэги
3. <script language="php">
   <?php print "Welcome"; ?>
   </script>
4. <% - тэги в стиле ASP

```
<?
$site_title = "PHP Recipies";
$bg_color = "white";
$user_name = "ivanov";
?>
<html>
<head>
<title> <? print $site_title; ?> <title>
</head>
<body bgcolor = "<? print $bg_color; ?>">
<?
//Присутствие тэгов HTML  в команде
print "<h3>PHP/HTML integration</h3>"
<?
</body>
</html>
```



Выражения , операторы и управляющие конструкции

Elseif

If (выражение) {блок}
elseif (выражение){блок}

While

While (выражение):
  Блок
Endwhile
(выход -> выражение -> ложь)

```
$n=5;
$ncopy=$n;
$factorial=1;
while($n>0):
  $factorial = $n*$factorial;
  $n--;
endwhile;
print “The factorial of $ncopy is
$factorial”
```

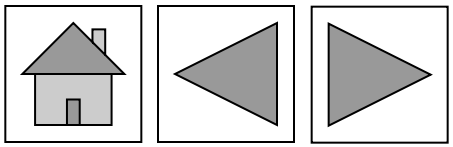
Do … While

```
Do:
  {Блок}
While (выражение);
```

```
$n=5;
$ncopy=$n;
$factorial=1;
do {
  $factorial = $n*$factorial;
  $n--;
}while($n>0)
print “The factorial of $ncopy is
$factorial”
```

For

```
for (инициализация; условие; приращение){
        блок}
```



Foreach

```
Foreach(массив as $элемент){
  блок
}
```

```
$menu=array(“pasta”,”steak”, “potatoes”,”fish”)
foreach($menu as $item {
print “$item<BR>”;
}
```

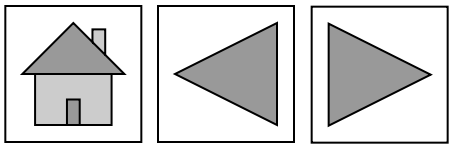
```
Foreach(массив as $ключ =>элемент ){
  блок
}
```

Switch

```
Switch (выражение)
Case(условие):
блок
Case(условие):
блок
        ...
        ...
default:
блок
}
```

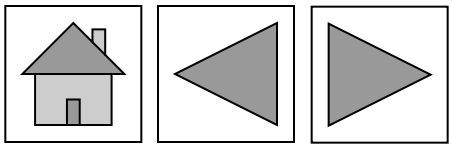
```
switch($user_input)
  case(“search”):
  print “search”;
  break;
  case(“dictionary”):
  print “dictionary”;
  break;
default:
  print “here is the menu…”;
  break;
}
```

Break -> прерывает выполнение конструкций while , for или switch
Continue -> пропускаются оставшиеся команды циклаи начинаются новые итерации



Массивы

- Индексируемые → $имя [индекс]; →
  $meat[0]=”chicken”
  $meat[1]=”steak”
  $meat[2]=”turkey”
- Ассоциативные → $имя [ключ]; →
  $languages[“spain”]=”spanish”
  $languages[“france”]=”french”



Массивы

Создание массивов

- Array
  ```
  $languages=array(“english”,”spanish”)
  $languages=array(“United States”=>”english”, “Spain”=>”spanish”)
  ```
- List
  ```
  list($name,$ocupation,$color)=split(“I”,$line)
  ```
- Range
  ```
  $lottery=range(0,9);
  ```

Поиск элемента в массиве

- In_array
  ```
  $languages=array(“english”,”spanish”);
  $exists=in_array(“russian”,”$languages”);//FALSE
  $exists=in_array(“english”,$languages”);//TRUE
  ```
- Array_keys
  ```
  $languages=array(“US”=>”english”; “Spain”=>”spanish”)
  $languages_keys=array_keys($languages);
  ```
- Array_values
  ```
  $languages=array(“US”=>”english”; “Spain”=>”spanish”)
  $languages_value=array_value($languages);
  ```

Добавление элементов в массив

- Array_push
  ```
  $languages=array(“spanish”);
  array_push($languages,”russian”);
  ```

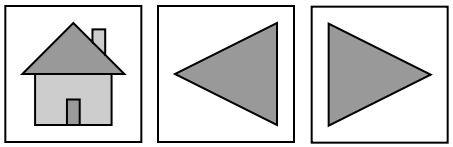
Удаление последнего элемента

- Array_pop
  ```
  a_languages=array_pop($languages);
  ```

Sizeof( ) - количество элементов в массиве

Sort( ) — сортировка

Rsort( ) — элементов массива



Файловый ввод/вывод

Проверка существования файлов

- file_exists

```
if(!file_exists ($filename)):
   print”File $filename does not exist!”;
endif;
```

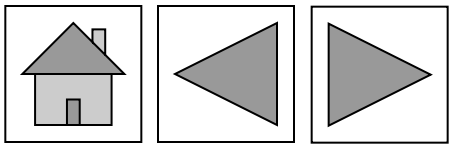
- is_file (проверяет кроме существования операции чтения/записи)

```
$file=”somefile.txt”;
if(is_file($file)):
   print”The file $file is valid and exists!”;
else:
   print”The file $file does not exist or it is not a valid file!”;
endif;
```

Открытие и закрытие файлов → fopen → fopen(файл,режим)

Файл
- имя локального файла
- php(стандартный поток I/O)
- http(подключение http к серверу)
- ftp(подключение ftp к серверу)

Режим
- r только чтение. Указатель в начало
- r+ чтение и запись. Указатель в начало
- w только запись. Указатель в начало содержимое файла уничтожается
- w+ чтение и запись. Указатель в начало содержимое файла уничтожается
- a только запись. Указатель в конец
- a+ чтение и запись. Указатель в конец



Открытие и закрытие файлов → fclose →

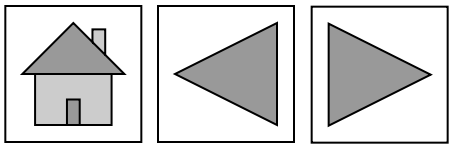
```
$file=”userdata.txt”;
if(file_exists($file)):
  $th=fopen($file,”r”);
  …
  fclose($th);
else:
  print”File $file does not exist!”;
endif;
```

Запись в файл → fwrite is_writeable (существует и разрешена запись) → fwrite(манипулятор,переменная) →

```
$data=”08:13:00”
$filename=”somefile.txt”
if(is_writeable($filename)):
$th=fopen($filename,”a+”);
$success=fwrite($th,$data);
fclose($th);
else:
print ”could not open
$filename for writing”
endif;
```

Чтение из файла → fread is_readable (существует и разрешено чтение) → fread(манипулятор,длина[в байтах]) →

```
$th=fopen(‘pastry.txt’,”r”);
file=fread($th,filesize($th));
print $file
fclose($th);
```



Работа с файловой системой

copy → copy(источник,приемник)

rename → rename(старое_имя,новое_имя)

unlink
(удаление файла) → unlink(файл)

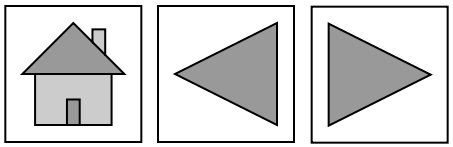
basename
(извлекает имя файла из полного имени) → basename(полное_имя) →
```
$path=”/usr/local/index.php”;
$title=basename($path);
```

dirname
(извлекает путь из полного имени файла) → dirname(путь) →
```
$path=”/usr/local/index.php”;
$file=dirname($path);
```

is_dir
(файл -> каталог?) → is_dir(имя_файла) →
```
$isdir =is_dir(“insex.htm”);//FALSE
$isdir =is_dir(“book”);//TRUE
```

opendir → opendir(путь) → открывает манипулятор для работы с каталогом

closedir → closedir(манипулятор) → закрывает манипулятор работы с каталогом



readdir
(возвращает очередной элемент каталога)

readdir(манипулятор_каталога)

```
$path=”/usr/local/index.php”;
$title=basename($path);
```

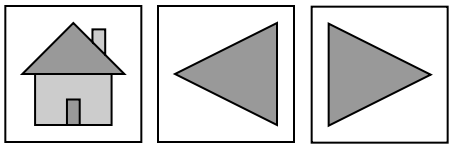
chdir
(переход в каталог)

chdir(каталог)

```
$newdir=”book”;
chdir($newdir)
$dh=opendir(‘.’);
print”files:”;
while ($file=readdir($dh)):
   print(“$file<br>”);
endwhile;
closedir($dh);
```

rewinddir
(переводит указатель текущей позиции в начало каталога, открытого функцией opendir( ))

rewinddir(манипулятор_каталога)



включение файлов
в сценарий PHP

include → include(файл) →

```
if(some_condition):
  include(‘text91a.txt’);
else:
  include(‘text91b.txt’);
endif;
```

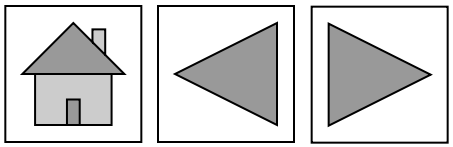
include_once → include_once(файл)

(проверяет , не был ли
он включен ранее ,
если файл уже был
включен ранее,
то вызов игнорируется)

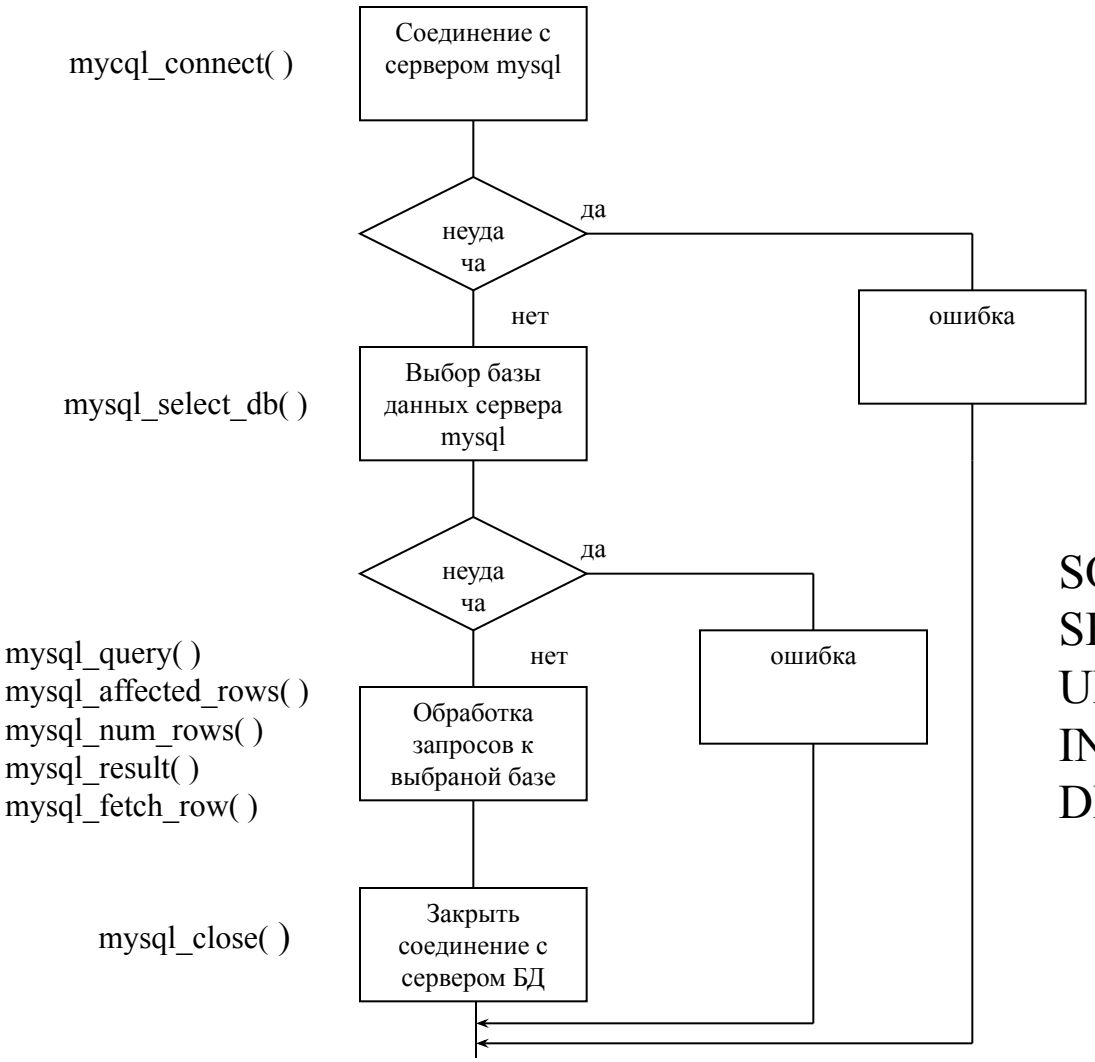
require → require(файл)

(включается всегда,
даже если стоит
в ветке “ложь”)

require_once(файл)



Базы данных



SQL
SELECT
UPDATE
INSERT
DELETE

mysql_connect

mysql_connect(хост,[имя_пользователя],[пароль])

Хост-имя хостового компьютера (по умолчанию локальныйкомпьютер)

имя_пользователя, пароль — в соответсвии с таблицами привелегий MYSQL

mysql_connect (“localhost”,”web”,”4tf9222f”)

mysql_select_db

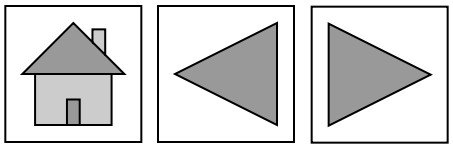
mysql_select_db(имя_базы_данных,[идентификатор_соединения])

(обязательный для более одной открытой БД)

mysql_select_db(”company”)

mysql_close

mysql_close([идентификатор_соединения])



mysql_query

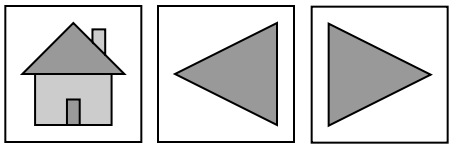
mysql_query(запрос,[идентификатор_соединения])

Запрос – запрос текста на SQL. Возвращает 0 в случае ошибки.
При отсутствии идентификатора_соединения запрос передается последнему открытому соединению
!!!mysql_query – не выполняет запрос и работает совместно с mysql_result и mysql_affected_rows

mysql_affected_rows

mysql_affected_rows([идентификатор_соединения])
Определяет количество записей в запросе SQL
с командами INSERT , UPDATE , REPLACE и DELETE
!не работает с SELECT

```
$querry = ”UPDATE products SET prod_name=\”cantaloupe\”
                WHERE prod_id=\”10001pr\””;
$result = mysql_query($query);
print ”Total row updated:”. mysql_affected_rows( );
```



mysql_num_rows → mysql_num_rows(результат) →

Количество записей в команде SELECT

```
$query=”SELECT prod_name
FROM products WHERE …”
$result=mysql_query($query);
Print”Total rows selected:
”.mysql_num_rows($result);
```

mysql_result → mysql_result(идентификатор_результата),запись,[поле])

получает результат SQL запроса

смещение в таблице

поле имя_поля

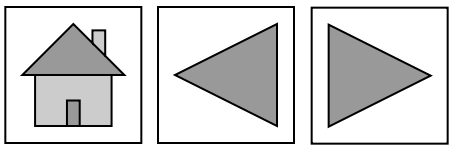
имя_поля.имя_таблицы

```
$query=”SELECT * FROM products”
$result=mysql_query($query);
$id=mysql_result($result,$x,’prod_id’);
$name=mysql_result($result,$x,’prod_name’);
$price=mysql_result($result,$x,’prod_price’);
```

mysql_fetch_row → mysql_fetch_row(результат) →

```
$query=”SELECT * FROM products”;
$result=mysql_query($query);
while
($row=mysql_fetch_row($result)):
print(“$row[“prod_id”]);
print(“$row[“prod_name”]);
print(“$row[“prod_price”]);
```

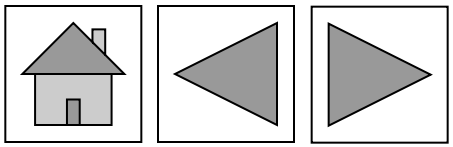
mysql_fetch_array - для ассоциативного массива.
Возвращает ассоциативный массив.



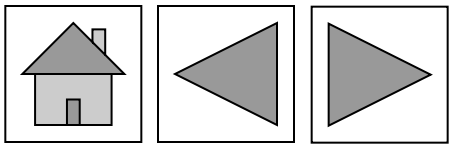
Usability

Категории

- Проблема первой категории, которая утратила первоначальную актуальность. Подобных ошибок при дизайне сайта желательно избегать, но их присутствие не приведет к краху сайта.
- Проблема второй категории и средней важности. При создании сайтов крайне желательно избегать таких проблем, но они больше не имеют первоочередного значения.
- Проблема третей категории. Остается весьма актуальной. Очень важно, чтобы дизайнеры продолжали уделять ей самое пристальное внимание.



Категория	Проблема
3	Ссылки, которые не изменяют цвет после их использования
3	Невозможность отмены выполненных действий
3	Открытие нового диалогового окна браузера
3	Всплывающее меню
3	Элементы дизайна, похожие на рекламу
3	Нарушение основных принципов и традиций дизайна
3	Странички с отсутствием понятного или полезного содержимого
3	Страницы, перенасыщенные текстом
1	Низкая скорость загрузки
1	Кадры
2	Флеш технологии
2	Малоэффективные результаты поиска
2	Мультимедия и видео
2	Фиксированная ширина страницы
2	Отсутствие межплатформенности
1	Неуверенные щелчки
2	Прокрутка страниц
1	Регистрация



Категория	Проблема
1	Регистрация
2	Сложные url адреса
1	Раскрывающиеся и каскадные меню
1	Подключаемые модули и новые технологии
1	Пользовательский интерфейс, созданный с помощью 3D технологий
1	Перенасыщенный дизайн
1	Заставки
1	Изменяющиеся графические изображения и прокручивание текста
2	Нестандартные элементы управления графическим интерфейсом пользователя
1	Отсутствие данных о поставщике информации
1	Придуманные термины
2	Устаревшее содержимое
2	Непоследовательная подача материала на сайте
2	Преждевременные требования о вводе конфиденциальной информации
2	Многочисленные сайты одной компании