1. Введение. Основные понятия

1.1. Internet

Интернет (сущ., м. р., 2 скл., с заглавной буквы). Всемирная общедоступная система соединенных между собой компьютерных сетей, предназначенная для передачи данных.

Конец 1960-х: ARPANET. [Defense] Advanced Research Projects Agency ([D]ARPA). Переход от *circuit switching* к *packet switching*.

1984: первая TCP/IP сеть по проекту National Science Foundation (NSF).

http://en.wikipedia.org/wiki/Internet

1.2. Intranet

Локальная сеть для внутреннего использования в какой-либо организации. Строится на тех же идеях, что и Internet.

http://en.wikipedia.org/wiki/Intranet

1.3. Hypertext

Парадигма пользовательского интерфейса: документы с гиперссылками. Автор идеи — Theodor Holm Nelson при участии Douglas Engelbart (1965).

Hypermedia — логическое развитие hypertext, включающее мультимедийные элементы (аудио, видео).

http://en.wikipedia.org/wiki/Hypertext
http://en.wikipedia.org/wiki/Ted_Nelson
http://en.wikipedia.org/wiki/Douglas_Engelbart
http://en.wikipedia.org/wiki/Hypermedia

1.4. WWW

World Wide Web. Совокупность ресурсов, идентифицируемых URL и связанных гиперссылками. Строится поверх Internet.

Автор идеи — Tim Berners-Lee из CERN (Centre Européen pour la Recherche Nucléaire). В 1990 году создал первый web-сервер (httpd) и web-браузер (WorldWideWeb). Работал над стандартами URI, HTML, HTTP. Основатель и глава W3C.

Злобствования: World Wild Web, World Wide Wait.

http://en.wikipedia.org/wiki/World_Wide_Web
http://en.wikipedia.org/wiki/Tim_Berners-Lee

1.5. URI/URL/URN

Uniform Resource Identifier. RFC 3986 (January 2005)

Uniform Resource Locator. RFC 1738 (December 1994)

Uniform Resource Name. RFC 2141 (May 1997)

URN идентифицирует некий ресурс (например `urn:ietf:rfc:1738`), а URL говорит, как этот ресурс найти (`http://www.ietf.org/rfc/rfc1738.txt`).

Общая схема построения URL:

```
scheme://user:pass@domain.tld:port/
dir/file.ext?param1=value1&param2=value2#anchor
```

Специальные и не-ASCII символы кодируются UTF-8 и побайтно записываются как `%XX`.
Абсолютные и относительные URL.
http://en.wikipedia.org/wiki/Uniform_Resource_Identifier
http://en.wikipedia.org/wiki/Uniform_Resource_Locator
http://en.wikipedia.org/wiki/Uniform_Resource_Name

1.6. HTML

HyperText Markup Language. Вырос из SGML (Standard Generalized Markup Language). Подробное обсуждение впереди.
http://en.wikipedia.org/wiki/HTML

1.7. HTTP

HyperText Transfer Protocol

- HTTP/0.9
- HTTP/1.0: RFC 1945 (May 1996)
- HTTP/1.1: RFC 2616 (June 1999)

При участии W3C, Microsoft, Xerox, Compaq.
Запросы GET, HEAD, POST, ...
Ответы:

- `200 OK`
- `403 Forbidden`
- `404 Not Found`
- `500 Internal Sever Error`

http://en.wikipedia.org/wiki/Hypertext_transfer_protocol

1.8. MIME

Multipurpose Internet Mail Extensions. Стандарт, определяющий способ передачи произвольных данных по e-mail. (SMTP поддерживает только 7-битные ASCII-тексты.) RFCs 2045–2049 (November 1996).
http://en.wikipedia.org/wiki/MIME

1.9. Domain name

Имя, отличающее компьютер от других машин в сети. Преобразуется в IP-адрес при помощи службы DNS.
Top-level domains (TLDs):

- роды деятельности: .com, .net, .org, .info, ...
- национальная принадлежность: .uk, .it, .jp, .ru, .su, ...

Domain hacks: del.icio.us, blo.gs, j@m.es, programme.ru, anekdotov.net.
http://en.wikipedia.org/wiki/Domain_name

1.10. RFC

Request For Comments. Информационные документы и стандарты, издаваемые IETF.
Пример шуточных RFC к 1 апреля:

- RFC 1149: IP over Avian Carriers.
- RFC 1217: Ultra Low-Speed Networking.
- RFC 2324: Hyper Text Coffee Pot Control Protocol.

http://en.wikipedia.org/wiki/Request_for_Comments
http://en.wikipedia.org/wiki/April_1st_RFC

1.11. IETF

Internet Engineering Task Force. Открытое международное сообщество, разрабатывающее и продвигающее стандарты для Internet.

Существует с 1986 года в виде регулярных собраний американских ученых. В 1990-х сменило статус с правительственной инициативы на независимую международную организацию под руководством ISOC.

http://en.wikipedia.org/wiki/Internet_Engineering_Task_Force
http://www.ietf.org/overview.html

1.12. IESG

Internet Engineering Steering Group. Верхушка IETF.
http://en.wikipedia.org/wiki/Internet_Engineering_Steering_Group

1.13. W3C

World Wide Web Consortium. Разрабатывает стандарты (рекомендации) для WWW. Основан в 1994 году Tim Berners-Lee.

http://en.wikipedia.org/wiki/World_Wide_Web_Consortium
http://www.w3.org/

1.14. ISOC

Internet Society. Международная организация, призванная «развивать Internet на пользу людям всего мира». Официально сформирована в 1992 году как образовательная некоммерческая.

http://en.wikipedia.org/wiki/Internet_Society
http://www.isoc.org/

1.15. IANA

Internet Assigned Numbers Authority. Организация, надзирающая за распределением IP-адресов, top-level domains и protocol numbers.

http://en.wikipedia.org/wiki/Internet_Assigned_Numbers_Authority
http://www.iana.org/

1.16. ICANN

Internet Corporation for Assigned Names and Numbers. Некоммерческая корпорация, контролирующая, в частности, деятельность IANA. Создана в 1998 году.

http://en.wikipedia.org/wiki/Internet_Corporation_for_Assigned_Names_and_Numbers
http://www.icann.org/tr/english.html

1.17. ISO

(От греч. isos — равный.) International Organization for Standardization. Сформирована в 1947 г. Включает представителей национальных стандартизующих организаций. Тесно связана с IEC.

Некорректное использование: iso-образы (файловая система ISO 9660), iso-единицы (чувствительность фотопленки).

http://en.wikipedia.org/wiki/International_Organization_for_Standardization

1.18. IEC

International Electrotechnical Commission. Стандартизует электрооборудование. Основана в 1906 г. Ввела систему единиц Giorgi, позднее выливщуюся в SI (Système International d'Unités).

http://en.wikipedia.org/wiki/International_Electrotechnical_Commission
http://www.iec.ch/

1.19. ISO/IEC JTC1

ISO/IEC Joint Technical Committee 1.

1.20. ITU

International Telecommunication Union. ITU-T = ITU Telecommunication Standardization Sector. Координирует стандартизацию в области телекоммуникаций. До 1992 г. известна как CCITT (от фр. Comité Consultatif International Téléphonique et Télégraphique). Последний образован в 1960 г.

http://en.wikipedia.org/wiki/ITU-T
http://www.itu.int/ITU-T/

1.21. Ecma International

Международная ассоциация по стандартизации информационных и коммуникационных технологий, а также бытовой электроники. Основана в 1961 г под именем European Computer Manufacturers Association.

http://en.wikipedia.org/wiki/Ecma_International
http://www.ecma-international.org/

1.22. OSI

Open Systems Interconnection. Совместная инициатива ISO и ITU-T по унификации и стандартизации сетевых протоколов, начатая в 1982 г. и свернутая в 1996 г.

http://en.wikipedia.org/wiki/Open_Systems_Interconnection

1.23. Модель OSI

OSI Reference Model. Абстрактная модель организации связи по компьютерным сетям. Состоит из семи слоев (уровней).

Стек протоколов: уровень предоставляет функции только вышележащему и пользуется функциями только нижележащего.

1. **Physical layer.** Электрические и физические спецификации устройств. Непосредственная передача бинарных данных (при необходимости модулированных) по каналу связи. На этом уровне: хабы.
2. **Data link layer.** Способ передачи данных непосредственным соседям по сети, возможность исправления ошибок. MAC-адресация. На этом уровне: свичи.
3. **Network layer.** Способ передачи последовательностей произвольной длины, маршрутизация данных к месту назначения, контроль ошибок. IP-адресация. На этом уровне: роутеры.
4. **Transport layer.** Предоставляет прозрачный способ передачи данных между пользователями. Пример: протокол TCP (Transmission Control Protocol).
5. **Session layer.** Управляет диалогом между пользовательскими программами. Устанавливает и разрывает соединения.
6. **Presentation layer.** Преобразование данных в пригодный для передачи вид.
7. **Application layer.** Наша программа.

Уровни 8 и 9: Financial, Political. Вариант: Money, Politics, Religion.
http://en.wikipedia.org/wiki/OSI_reference_model

2. Разметка web-страниц. Язык HTML

- HTML 2.0 — 1995 год.
- HTML 3.0 — 1995 год.
- HTML 3.2 — 1997 год.
- HTML 4.0 — 1997 год.
- HTML 4.01 — 1999 год.

Ссылки:

- http://www.w3.org/TR/html4/
- http://www.w3.org/People/Raggett/tidy/
- http://validator.w3.org/

2.1. Простейший пример

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
    <title>Page title</title>
</head>
<body>
    <p>hello world</p>
</body>
</html>
```

2.2. Синтаксис языка

Элемент состоит из:

- открывающий тег с атрибутами (иногда опционально: `html`);
- содержимое (иногда опционально);
- закрывающий тег (иногда опционально).

Регистр тегов и атрибутов не важен, но будем писать все в нижнем регистре, чтобы потом перейти к XHTML. Последовательность пробелов и переводов строк сжимается до одного пробела (кроме случая `pre`).

Атрибуты заключаются в двойные или одинарные кавычки. Для алфавитно-цифровых значений кавычки не обязательны, но рекомендуются.

Особый случай — булевские атрибуты.

Символы вводятся:

- непосредственно (`a`, `b`);
- по имени (`&` `<` `>` `&quot;`);
- по коду (`&#229;` `&#x6C34;`).

Комментарии: `<!-- comments -->`.

2.3. DTD

DTDs:

- `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">`
- `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">`
- `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">`

2.4. Основные элементы и их атрибуты

Важнейшие элементы: `html`; `head`, `title`, `meta`; `body`.
Элементы разделяются на:

- блочные (block-level); могут содержать текст и элементы любого типа, всегда начинают новую строку;
- строчные (inline, text-level); могут содержать только текст и строчные элементы, новую строку не начинают.

Атрибуты почти всех элементов: `lang`, `dir`, `id`, `class`, `title`.
Обработчики событий: `onload`, `onunload`, `onclick`, `onmousemove`, ...
Атрибуты типа `color`/`bgcolor` трогать не будем. Это — территория CSS.
Отступление о -, `&ndash;`, `&mdash;`, `&minus;`.

2.5. Заголовки

`h1`, ..., `h6`. Внимание к правильной организации структуры заголовков.

2.6. Абзацы, строки, блоки

Абзац `p`, перевод строки `br`.
Блоки `div`, `span`.

2.7. Форматированный текст

`pre`

2.8. Гиперссылки

`a`: атрибуты `id`/`name`, `href`.

2.9. Логическая разметка

`abbr`, `acronym`, `address`, `cite`, `code`, `dfn`, `em`, `kbd`, `samp`, `strong`, `var`.

2.10. Цитирование

`blockquote`, `q`: атрибут `cite`.

2.11. Модификации документа

`ins`, `del`; атрибуты `datetime` (в формате `YYYY-MM-DDThh:mm:ssTZD`), `cite`.

2.12. Индексы

`sub`, `sup`.

2.13. Списки

Списки: `ol`, `ul`, `dl`.

2.14. Таблицы

`table`. Важные атрибуты:

- `summary`;
- `frame`=`void|above|below|hsides|lhs|rhs|vsides|box|border`;
- `rules`=`none|groups|rows|cols|all`.

Содержимое:

- `caption`;
- `colgroup`, `col`;
- `thead`, `tfoot`, `tbody`;
- ячейки: `th`, `td`;
  - атрибуты `abbr`, `cospan`, `rowspan`,
  - `scope`=`col|row|colgroup|rowgroup`.

2.15. Картинки, апплеты и проч.

`img`: `src`, `alt`, `longdesc`.
`applet`: deprecated.
`object`: `data`, `type`.

2.16. Карты

`map` с атрибутом `name`. Подключение карты осуществляется при помощи атрибута `usemap` у элементов `a`, `object`, `input`.

Внутри `map`: `area` и/или `a` с атрибутами `href`, `alt`, `shape`=`default|rect|circle|poly` и `coords`.

При этом все внутренности `map`, кроме `area`, будут отображены.

Разбор полетов по результатам практики

Регистр заголовков: обычный. В верхний регистр лучше переводить средствами CSS. Пример неправильной интерпретации верхнего регистра экранной читалкой — CONTACT US.

Отступ при помощи ` ` или `blockquote` грозит отрывом рук. Лучше тогда не делать вообще.

Еще раз о структуре заголовков. Первый заголовок страницы всегда `h1`. Далее — либо `h1` (следующий равнозначный раздел), либо `h2` (вложенный подраздел).

Нельзя вкладывать блочные элементы внутрь `p`. Если нужен список или цитата, сначала закрываем `p`, потом начинаем соответствующий элемент. Кстати, внутри `blockquote` текст тоже надо организовывать в абзацы.

Можно делать перекрестные ссылки внутри документа: `id="myid"` и `<a href="#myid">`.

Выразительных средств HTML хватает далеко не для каждого документа. В таком случае помогают `div` и `span`, не нагруженные семантикой.

3. Семейство языков XHTML

3.1. XHTML 1.0

XHTML 1.0 — HTML 4.01 в синтаксисе XML. Основные изменения:

- теги, атрибуты и значения в нижнем регистре;
- все значения атрибутов в кавычках;
- все элементы имеют открывающий и закрывающий тег, либо имеют вид `<br />`;
- `Content-Type: application/xhtml+xml`;
- `<html xmlns="http://www.w3.org/1999/xhtml">`;
- рекомендуется `<?xml version="1.0" encoding="UTF-8"?>`.

Преимущества:

- Наведение строгости и ликвидация неоднозначностей, возникавших в HTML из-за возможности опускания закрывающих тегов.
- Возможность использования средств, работающих с XML.
- Возможность внедрения XHTML в XML и наоборот (SVG, MathML).

DTDs:

- ```
  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
  ```
- ```
  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
  ```
- ```
  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
  ```
  ```

3.2. XHTML 1.1

XHTML 1.1 — переработанный с учетом модуляризации XHTML 1.0 Strict.
Отличия:

- ликвидация deprecated элементов и атрибутов;
- атрибут `lang` заменен на `xml:lang`;
- атрибут `name` заменен на `id` у элементов `a` и `map`.

DTD:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
    "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

3.3. XHTML Basic

Подмножество модулей XHTML 1.1, часть из них упрощены. Упор на маленькие устройства, которые не потянут полную версию XHTML 1.1.
DTD:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
    "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
```

4. Настройка вида web-страницы. Язык CSS

Cascading Style Sheets — каскадные таблицы (листы) стилей.

- level 1: 1996, revised 1999
- level 2: 1998
- level 2.1: working draft 2005

Ссылки:

- http://www.w3.org/TR/CSS21/
- http://jigsaw.w3.org/css-validator/
- http://csszengarden.com/

4.1. Подключение таблицы стилей к документу

- ```
  <link rel="stylesheet" type="text/css" href="main.css"/>
  <link rel="alternate stylesheet" type="text/css" href="alt.css"/>
  ```
- ```
  <style type="text/css">
  ...
  </style>
  ```
- ```
  <style type="text/css">
  @import url("style.css");
  </style>
  ```
- ```
  <elem style="...">...</elem>
  ```
- ```
  <?xml-stylesheet type="text/css" href="main.css"?>
  <?xml-stylesheet alternate="yes" type="text/css" href="alt.css"?>
  ```

4.2. Простейший пример

```
h1 {
    font-size: 20px;
    color: orange;
}
h2 {
    font-size: 18px;
    color: orange;
}
```

4.3. Структура стилевого файла

```
selector {
    /* comment */
    property: value;
    ...
}
...
```
```

Ввод символов:

- непосредственный (`a`, `b`, ...)
- при помощи кодов (`\0`–`\FFFFFF`)
- при помощи `\` (`\?`, `\>`, ...)

Возможен разрыв строки внутри кавычек с использованием `\`.

4.4. Единицы измерения

Относительные:

- `em` = `font-size`
- `ex` = `x-height`
- `px`

Абсолютные:

- `in` (дюйм) = 2.54 см
- `cm` (сантиметр)
- `mm` (миллиметр)
- `pt` (пункт) = $\frac{1}{72}$ дюйма
- `pc` (пика) = 12 пунктов

4.5. Способы задания цвета

- `#xyz`
- `#zzyyzz`
- `rgb(x,y,z)`
- `rgb(x%,y%,z%)`

4.6. Селекторы

- `*` — universal selector, во многих случаях можно опустить
- `a` — type selector, `a` — название элемента
- `a b` — descendant selector
- `a > b` — child selector
- `a + b` — adjacent sibling selector
- `a[att]`, `a[att=val]`, `a[att~=val]`, `a[att|=val]` — attribute selectors
- `a.b` — class selector (только в HTML)
- `a#b` — id selector

4.7. Псевдоклассы

- `:first-child`
- `:link`, `:visited` (для гиперссылок)
- `:hover`, `:active`, `:focus`
- `:lang(a)`

4.8. Псевдоэлементы

- `:first-line` (для блочных элементов, строчные — ограниченно)
- `:first-letter` (только для блочных элементов)

4.9. Наследование свойств

Многие свойства наследуются вложенными элементами. Остальные можно унаследовать, указав значение `inherit`.

Источники стилей (в порядке уменьшения веса):

- авторская таблица стилей;
- пользовательская таблица стилей;
- таблица стилей, зашитая в браузер.

Ключевое слово `!important` добавляет вес правилу.

4.10. Выходные устройства

Устройство, для которого предназначена таблица стилей, задается одним из следующих способов:

- `@import url("sp.css") screen, print;`
- ```
  @media projection {
      ...
  }
  ```
- `<link rel="stylesheet" type="text/css" href="main.css" media="all"/>`

Типы устройств: `all`, `aural`, `braille`, `embossed`, `handheld`, `print`, `projection`, `screen`, `tty`, `tv`.

4.11. Боксовая модель

Каждый элемент состоит из контента (content), отступа (padding), границы (border) и полей (margin).

Верхнее/нижнее поля идущих следом элементов могут сливаться (collapse). Размером результирующего поля назначается максимум исходных полей.

Свойства:

- `width`, `height`;
- `padding-top`, `padding-right`, `padding-bottom`, `padding-bottom`, `padding`;
  ```

- margin-top, margin-right, margin-bottom, margin-bottom, margin;
- border-*-width, border-width (thin, medium, thick, размер);
- border-*-color, border-color (transparent, цвет);
- border-*-style, border-style:
  - none;
  - hidden (особо ведет себя в таблицах);
  - dotted;
  - dashed;
  - solid;
  - double;
  - groove;
  - ridge;
  - inset;
  - outset.

4.12. Цвет и фон

Фон покрывает области контента и отступа. Поля всегда прозрачные.

- color;
- background-color;
- background-image: url(...);
- background-repeat: repeat, repeat-x, repeat-y, no-repeat;
- background-attachment: fixed, scroll;
- background-position: top, center, bottom, left, center, right, проценты, размеры;
- background.

Дополнительные значения цвета: ActiveBorder, ActiveCaption, AppWorkspace, Background, ButtonFace, ButtonHighlight, ButtonShadow, ButtonText, CaptionText, GrayText, Highlight, HighlightText, InactiveBorder, InactiveCaption, InactiveCaptionText, InfoBackground, InfoText, Menu, MenuText, Scrollbar, ThreeDDarkShadow, ThreeDFace, ThreeDHighlight, ThreeDLightShadow, ThreeDShadow, Window, WindowFrame, WindowText.

4.13. Текст

- text-indent;
- text-align: left, right, center, justify;
- text-decoration: none, underline, overline, line-through, blink;
- text-transform: capitalize, uppercase, lowercase, none;
- white-space: normal, pre, nowrap;
- letter-spacing, word-spacing: normal, размер;
- line-height: normal, размер.

4.14. Шрифты

- font-family: serif, sans-serif, cursive, fantasy, monospace;
- font-style: normal, oblique, italic;
- font-variant: normal, small-caps;
- font-weight: 100–900, normal, bold, bolder, lighter;
- font-stretch: normal, wider, narrower, ultra-condensed, extra-condensed, condensed, semi-condensed, semi-expanded, expanded, extra-expanded, ultra-expanded;
- font-size: larger, smaller, xx-small, x-small, small, medium, large, x-large, xx-large, размер;
- font.

4.15. Позиционирование и отображение

- position: static, relative, absolute (top, right, bottom, left), fixed;
- float: left, right, none;
- clear: left, right, both, none;
- display: block, inline, none, ...;
- visibility: visible, hidden, collapse;
- z-index: auto, число;
- vertical-align: baseline, sub, super, top, text-top, middle, bottom, text-bottom;
- overflow: visible, hidden, scroll.

4.16. Списки

- list-style-type: disc, circle, square, decimal, decimal-leading-zero, lower-roman, upper-roman, lower-greek, lower-alpha, lower-latin, upper-alpha, upper-latin, hebrew, armenian, georgian, cjk-ideographic, hiragana, katakana, hiragana-iroha, katakana-iroha, none;
- list-style-image: url(...);
- list-style-position: inside, outside,
- list-style.

4.17. Таблицы

- border-collapse: collapse, separate;
- border-spacing: размеры;
- empty-cells: show, hide.

4.18. Курсор

`cursor`: `url(...)`, `auto`, `crosshair`, `default`, `pointer`, `move`, `e-resize`, `ne-resize`, `nw-resize`, `n-resize`, `se-resize`, `sw-resize`, `s-resize`, `w-resize`, `text`, `wait`, `help`.

Задачи для разбора

- картинки внутри гиперссылок снабдить рамками;
- отцентровать заголовочные ячейки, относящиеся к столбцу, и выровнять по правому краю относящиеся к строке;
- создать блок (как в новостях на CAT);
- абзацы, расположенные вне таблиц, снабдить абзацными отступами;
- создать буквицы у абзацев вне таблиц;
- создать пунктирное подчеркивание для `acronym`, у которых указан атрибут `title`;
- уменьшить расстояние перед первым заголовком и между смежными заголовками;
- выделить каждый абзац внутри цитаты линией слева.

5. Интерактивные элементы страниц. Язык JavaScript

- 1995 — язык JavaScript представлен публике компаниями Sun Microsystems и Netscape Communications Corporation. Автор — Brendan Eich.
- Март 1996 — Netscape Navigator 2.0 с поддержкой JavaScript.
- Август 1996 — Internet Explorer 3.0 с языком JScript.
- Июнь 1997 — первая редакция стандарта ECMA-262.
- Июнь 1998 — вторая редакция.
- Декабрь 1999 — третья редакция.
- Июнь 2004 — стандарт E4X (ECMAScript for XML): ECMA-357.

Ссылки:

- http://developer.mozilla.org/en/docs/JavaScript
- http://www.ecma-international.org/publications/standards/Ecma-262.htm

5.1. Подключение скрипта к документу

- `script`: `type`, `src`;
- свойства `onload`, `onunload`, `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onfocus`, `onblur`, `onkeypress`, `onkeydown`, `onkeyup`, `onsubmit`, `onreset`, `onselect`, `onchange`;
- `<a href="javascript:...">click here</a>`.

5.2. Простейший пример

```
<html>
<head>
<title>JavaScript example</title>
<script type="text/javascript">
function hello() {
    alert("hello world!");
}
</script>
</head>
<body onload="hello()">
<p>JavaScript example</p>
</body>
</html>
```

5.3. Синтаксис языка

```
/*  multiline
    comment     */
statement;
statement;  // comment
```

Внимание: semicolon insertion.

5.4. Переменные

Переменные типизируются динамически. Объявление переменной:

- `var name = value;`
- `name = value;`

Переменная, объявленная с `var` внутри функции, является локальной для этой функции.
Примитивные типы:

- `Undefined`: `undefined`;
- `Null`: `null`;
- `Boolean`: `true`, `false`;
- `Number`: десятичное или шестнадцатеричное (`0x`) число, `NaN`, `Infinity`;
- `String`: строка в одинарных или двойных кавычках.

Escape-sequences: `\t`, `\n`, `\r`, `\\`, `\v`, `\f`, `\b`, `\uXXXX`.
Непримитивный тип: `Object`.
Встроенные объекты: `Global`, `Object`, `Function`, `Array`, `Boolean`, `Date`, `Math`, `Number`, `RegExp`, `String`, `Error`, `EvalError`, `RangeError`, `ReferenceError`, `SyntaxError`, `TypeError`, `URIError`.

5.5. Операторы

Унарные операторы:

- постфиксные и префиксные `++` и `--`;
- `+`, `-`; `~`, `!`;
- `delete`;
- `void`;
- `typeof`;
- `in`;
- `instanceof`.

Бинарные операторы:

- `*`, `/`, `%`, `+`, `-`;
- `<<`, `>>`, `>>>`.

Операторы сравнения:

- `<`, `>`, `<=`, `>=`;
- `==`, `!=`, `===`, `!==`.

Побитные операторы:

- `&`, `|`, `^`.

Логические операторы:

- `&&`, `||`.

Операторы присваивания:

- `=`, `*=`, `/=`, `%=`, `+=`, `-=`;
- `<<=`, `>>=`, `>>>=`;
- `&=`, `^=`, `|=`.

5.6. Управляющие структуры

Условный оператор

```
if (condition) {
    statements
} else {
    statements
}

condition ? statement : statement;
```

Циклы

```
while (condition) {
    statements
}

do {
    statements
} while (condition);

for (initial-expression; condition; increment-expression) {
   statements
}

for (slot in object) {
    statements involving object[slot]
}
```

Операторы: `break`, `continue` (с необязательной меткой).

Множественный выбор

```
switch (expression) {
case val1 :
    statements;
    break;
case val2 :
    statements;
    break;
default :
    statements;
}
```

Присоединение к объекту

```
with (expression) {
    statements
}
```

Обработка исключений

```
try {
    statements
    throw expression
} catch (identifier) {
    statements
} finally {
    statements
}
```

5.7. Функции

```
function name(arg1, arg2, arg3) {
    statements;
    return expression;
}

var name = function(arg1, arg2, arg3) {
    statements;
    return expression;
}

var name = new Function("arg1", "arg2", "arg3", "statements");
```

Возможен доступ к параметрам через массив `arguments`.
Простые типы передаются по значению, объекты — по ссылке.

5.8. Массивы

Создание массива:

- `myArray = [0,1,,,4,5];`
- `myArray = new Array(0,1,2,3,4,5);`
- `myArray = new Array(365);`

Свойство `length`: максимальный индекс плюс 1.
Методы:

- `concat([item1[, item2[, ...]]]);`
- `join([separator]);`
- `pop();`
- `push([item1[, item2[, ...]]]);`
- `reverse();`

- `shift()`;
- `slice(start[, end])`;
- `sort([comparator])`;
- `splice(start, count[, item1[, item2[, ...]]])`;
- `unshift([item1[, item2[, ...]]])`.

5.9. Встроенные функции

- `eval(code)`;
- `parseInt(str, radix)`;
- `parseFloat(str)`;
- `isNaN(num)`;
- `isFinite(num)`;

5.10. Объект `Math`

Константы:

- `Math.E`
- `Math.PI`
- `Math.SQRT2`
- `Math.SQRT1_2`
- `Math.LN2`
- `Math.LN10`
- `Math.LOG2E`
- `Math.LOG10E`

Методы:

- `Math.abs(x)`
- `Math.acos(x)`
- `Math.asin(x)`
- `Math.atan(x)`
- `Math.atan2(y,x)`
- `Math.ceil(x)`
- `Math.cos(x)`
- `Math.exp(x)`
- `Math.floor(x)`

- `Math.log(x)`
- `Math.max(x,y[, z[, ...]])`
- `Math.min(x,y[, z[, ...]])`
- `Math.pow(x,y)`
- `Math.random()`
- `Math.round(x)`
- `Math.sin(x)`
- `Math.sqrt(x)`
- `Math.tan(x)`

5.11. Объекты `Number`

- `toString(radix)`
- `toFixed(digits)`
- `toExponential(digits)`
- `toPrecision(precision)`

5.12. Объекты `String`

- `charAt(i)`
- `charCodeAt(i)`
- `concat(s1[, s2[, ...]])`
- `indexOf(s[, i])`
- `lastIndexOf(s[, i])`
- `replace(old, new)`
- `slice(start, end)`
- `split(separator[, limit])`
- `substr(start, length)`
- `substring(start, end)`
- `toLowerCase()`
- `toUpperCase()`

5.13. Объект `Date`

- `new Date(year, month[, date[, ...[, ms]]]);`
- `Date.parse(value);`
- `toDateString();`
- `toTimeString();`
- `getTime()`, `setTime(time);`
- `getFullYear()`, `setFullYear(year[, month[, date]]);`
- `getMonth()`, `setMonth(month[, date]);`
- `getDate()`, `setDate(date);`
- `getDay();`
- `getHours()`, `setHours(hours[, ...]);`
- `getMinutes()`, `setMinutes(min[, sec[, ms]]);`
- `getSeconds()`, `setSeconds(sec[, ms]);`
- `getMilliseconds()`, `setMilliseconds(ms)`.

5.14. Объекты

Идеология JavaScript — prototype-based (instance-based, class-less) programming. Конструктором может быть любая функция, вызванная с `new`.

```
function Point(x, y) {
    this.x = x;
    this.y = y;
}
Point.prototype.abs = function() {
    return Math.sqrt(x * x + y * y);
}
var pt = new Point(3, 4);
var d = pt.abs();
```

5.15. DHTML

Dynamic HTML — комбинация HTML, таблиц стилей и скриптов (например, JavaScript), позволяющая анимировать документ.

Объекты: `window`, `document`, `location`, `history`.

Функции: `alert`, `confirm`, `prompt`.

Ссылки:

- http://developer.mozilla.org/en/docs/DHTML
- http://en.wikipedia.org/wiki/DHTML
- http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/dhtml_reference_entry.asp

5.16. AJAX

Asynchronous JavaScript And XML. Комбинация DHTML и с динамической подгрузкой данных с сервера без необходимости обновлять страницу целиком. Ключевой элемент:

```
var req = new XMLHttpRequest();
req.open('GET', 'http://www.example.com/data.xml', false);
req.send(null);
// process req.responseXML as XMLDocument
```

Ссылки:

- http://developer.mozilla.org/en/docs/AJAX
- http://en.wikipedia.org/wiki/AJAX

5.17. DOM

Document Object Model. Платформенно и языково независимый интерфейс, позволяющий программам работать с содержимым, структурой и стилем документов. Разрабатывается W3C.

- DOM Level 0 :)
- DOM Level 1: 1 October 1998;
- DOM Level 1 (Second Edition, Draft): 29 September 2000;
- DOM Level 2 Core: 13 November 2000;
- DOM Level 3 Core: 7 April 2004.

Ссылки:

- http://www.w3.org/DOM/DOMTR
- http://www.w3.org/2003/02/06-dom-support.html

Далее представлены наиболее важные интерфейсы DOM Level 1 (с сокращениями).

Node

```
interface Node {
  // NodeType
  const unsigned short ELEMENT_NODE                = 1;
  const unsigned short ATTRIBUTE_NODE              = 2;
  const unsigned short TEXT_NODE                   = 3;
  const unsigned short CDATA_SECTION_NODE          = 4;
  const unsigned short ENTITY_REFERENCE_NODE       = 5;
  const unsigned short ENTITY_NODE                 = 6;
  const unsigned short PROCESSING_INSTRUCTION_NODE = 7;
  const unsigned short COMMENT_NODE                = 8;
  const unsigned short DOCUMENT_NODE               = 9;
  const unsigned short DOCUMENT_TYPE_NODE          = 10;
  const unsigned short DOCUMENT_FRAGMENT_NODE      = 11;
  const unsigned short NOTATION_NODE               = 12;
```

```
  readonly attribute DOMString      nodeName;
           attribute DOMString      nodeValue;
  readonly attribute unsigned short nodeType;
  readonly attribute Node           parentNode;
  readonly attribute NodeList       childNodes;
  readonly attribute Node           firstChild;
  readonly attribute Node           lastChild;
  readonly attribute Node           previousSibling;
  readonly attribute Node           nextSibling;
  readonly attribute NamedNodeMap   attributes;
  readonly attribute Document       ownerDocument;

  Node      insertBefore(in Node newChild,
                         in Node refChild);
  Node      replaceChild(in Node newChild,
                         in Node oldChild);
  Node      removeChild(in Node oldChild);
  Node      appendChild(in Node newChild);
  boolean   hasChildNodes();
  Node      cloneNode(in boolean deep);
};
```

Document

```
interface Document : Node {
  readonly attribute DocumentType       doctype;
  readonly attribute DOMImplementation  implementation;
  readonly attribute Element            documentElement;

  Element               createElement(in DOMString tagName);
  DocumentFragment      createDocumentFragment();
  Text                  createTextNode(in DOMString data);
  Comment               createComment(in DOMString data);
  CDATASection          createCDATASection(in DOMString data);
  ProcessingInstruction createProcessingInstruction(in DOMString target,
                                                     in DOMString data);
  Attr                  createAttribute(in DOMString name);
  EntityReference       createEntityReference(in DOMString name);
  NodeList              getElementsByTagName(in DOMString tagname);
};
```

NodeList

```
interface NodeList {
  Node item(in unsigned long index);
  readonly attribute unsigned long length;
};
```

Element

```
interface Element : Node {
  readonly attribute  DOMString tagName;
```

```
  DOMString getAttribute(in DOMString name);
  void      setAttribute(in DOMString name,
                         in DOMString value);
  void      removeAttribute(in DOMString name);
  Attr      getAttributeNode(in DOMString name);
  Attr      setAttributeNode(in Attr newAttr);
  Attr      removeAttributeNode(in Attr oldAttr);
  NodeList  getElementsByTagName(in DOMString name);
  void      normalize();
};
```

HTMLDocument

```
interface HTMLDocument : Document {
           attribute DOMString      title;
  readonly attribute DOMString      referrer;
  readonly attribute DOMString      domain;
  readonly attribute DOMString      URL;
           attribute HTMLElement    body;
  readonly attribute HTMLCollection images;
  readonly attribute HTMLCollection applets;
  readonly attribute HTMLCollection links;
  readonly attribute HTMLCollection forms;
  readonly attribute HTMLCollection anchors;
           attribute DOMString      cookie;
  void      open();
  void      close();
  void      write(in DOMString text);
  void      writeln(in DOMString text);
  Element   getElementById(in DOMString elementId);
  NodeList  getElementsByName(in DOMString elementName);
};
```

HTMLElement

```
interface HTMLElement : Element {
  attribute DOMString id;
  attribute DOMString title;
  attribute DOMString lang;
  attribute DOMString dir;
  attribute DOMString className;
};
```

6. Введение в серверное программирование

6.1. HTML-формы

Изложение по стандарту HTML 4.01. Ссылки см. в соответствующем разделе.

Форма — элемент документа, содержащий наряду с обычной разметкой элементы управления: кнопки, меню и т. д. Позволяет отправлять данные на сервер.

Form

- `action` — URL получателя данных;
- `method` — `get` или `post`;
- `enctype`, `accept`, `name`, `onsubmit`, `onreset`, `accept-charset`.

Input

- `type`: `text`, `password`, `checkbox`, `radio`, `submit`, `reset`, `file`, `hidden`, `image`, `button`;
- `name`, `value`;
- `checked` (для checkbox и radio);
- `size`, `maxlength`;
- `src`, `alt` (для image).

Button

- `name`, `value`;
- `type`: `submit`, `button`, `reset`.

Select

- `name`;
- `size` — количество видимых строк;
- `multiple`, `disabled`.

Optgroup

`label`

Option

- `selected`;
- `label`;
- `value`.

Textarea

- `name`;
- `rows`, `cols`.

Структурные элементы

`fieldset`, `legend`
`label`: `for` — `id` элемента управления. Можно не указывать `for`, а включить соответствующий элемент внутрь `label`.

6.2. CGI

Common Gateway Interface. Стандарт на взаимодействие HTTP-сервера с внешними программами, запускаемыми сервером вместо чтения содержимого статического HTML-файла. Текущая версия — CGI 1.1.

Общая схема:

- получение сервером запроса пользователя;
- установка переменных окружения (`REQUEST_METHOD`, `QUERY_STRING`, `CONTENT_LENGTH`, ...);
- запуск внешней программы:
  - в случае `REQUEST_METHOD=POST` пользовательские данные подаются на стандартный вход программы;
  - в случае, если `QUERY_STRING` не содержит символа `=`, ее значение передается программе в командной строке;
- анализ вывода программы и возвращение ответа пользователю.

Пример обработчика GET-запроса:

```
@echo off
echo Content-type: text/plain
echo.
echo Request from %REMOTE_ADDR%
echo QUERY_STRING=%QUERY_STRING%
```

Ссылки:

- http://en.wikipedia.org/wiki/Common_Gateway_Interface
- http://hoohoo.ncsa.uiuc.edu/cgi/

6.3. SSI

Server Side Includes. Директивы, включаемые в HTML-файл и исполняемые на стороне сервера в момент отправки страницы клиенту.

Рассмотрим реализацию SSI в сервере Apache 2.0 (`mod_include`). Общий вид:

```
<!--#directive param="value" ... -->
```

Директивы:

- `config`: `timefmt` (синтаксис `strftime`), `sizefmt` (`abbrev`, `bytes`), `errmsg`;

- `echo`: `var`, `encoding`;
- `exec`: `cmd` или `cgi`;
- `fsize`: `file` или `virtual`;
- `flastmod`: `file` или `virtual`;
- `include`: `file` или `virtual`;
- `printenv`;
- `set`: `var`, `value`;
- `if`: `expr`; `elif`: `expr`; `else`; `endif`.

Внутри скрипта доступны CGI-переменные, а также некоторые другие, например, `LAST_MODIFIED`.
Ссылки:

- http://httpd.apache.org/docs/2.0/howto/ssi.html
- http://httpd.apache.org/docs/2.0/mod/mod_include.html

6.4. LAMP

Комбинация Linux, Apache, MySQL, Perl/PHP/Python, ставшая популярной платформой web-разработки благодаря бесплатности и доступности.
Ссылки:

- http://en.wikipedia.org/wiki/LAMP_(software_bundle)

7. Perl

Кроссплатформенный язык программирования. Создатель — Larry Wall. Одна из появившихся впоследствии расшифровок — Practical Extraction and Report Language.

- Perl 1.0 — 1987.
- Perl 2.0 — 1988.
- Perl 3.0 — 1989.
- Perl 4.0 — 1991.
- Perl 5.0 — 1994.
- ...
- Perl 5.8.7 — 3 June 2005.

Письменная спецификация языка не существует, ее написание для текущей версии не планируется. Стандартом служит реализация интерпретатора.

Ссылки:

- http://www.perl.org/
- http://www.cpan.org/
- http://en.wikipedia.org/wiki/Perl/
- http://activestate.com/Products/ActivePerl/

7.1. Синтаксис

```
#comment
statement;
=pod
Plain Old Documentation
=cut
```

7.2. Переменные

Имена переменных чувствительны к регистру. Имена переменных разных типов находятся в разных пространствах имен.

Scalar

Скаляр — строка, число или ссылка. Разумные преобразования выполняются автоматически.

```
my $a = 'a';
my $b = "$a\n";
my $c = 3.14;
```

Внутри строк в двойных кавычках производится «интерполяция».

Форматы чисел:

```
12345
12345.67
.23E-10             # a very small number
3.14_15_92          # a very important number
4_294_967_296       # underscore for legibility
0xff                # hex
0xdead_beef         # more hex
0377                # octal (only numbers, begins with 0)
0b011011            # binary
```

Array

Массив представляет список элементов: `my @a = ($a, $b, $c);`

Доступ к элементу массива: `$a[0]`. Отрицательные индексы адресуют элементы, начиная с конца массива.

Ссылка на массив:

```
my $a = [1, 2, 3];
```

Индекс последнего элемента: `$#a`.

Количество элементов массива: `scalar(@a)`.

Выборка элементов в новый массив:

```
@a[0,2];    # индексы элементов в нужном порядке
@a[0..1];   # диапазон индексов
```

Hash

Хеш — множество пар ключ/значение.

```
my %h1 = ("a", "b", "c", "d");
my %h2 = (a => "b", c => "d");
my @slice = @h1{"a", "c"};
```

Доступ к значению по ключу: `$h1{"a"}`.

Ссылка на хеш:

```
my $h = {a => b};
```

Получение отдельно ключей и значений:

```
my @keys = keys %h1;
my @values = values %h1;
```

Область видимости

Переменные без модификатора являются глобальными. Модификатор `my` ограничивает область видимости ближайшим блоком `{...}`.

Операторы

Аргументы функций можно не заключать в скобки. Однако если за именем функции следует открывающая скобка, то аргументом считается содержимое этих скобок.

```
print (1 + 2) + 3; # prints 3
print 1 + (2 + 3); # prints 6
print +(1 + 2) + 3; # prints 6
```

Инкремент и декремент: `++` и `--` (префиксная и постфиксная формы); дополнительная возможность инкремента — работа со строками, удовлетворяющими регулярному выражению `/^[a-zA-Z]*[0-9]*/`.

Возведение в степень `**`.

Унарные операторы

- `!` — логическое отрицание;
- `not` — логическое отрицание с меньшим приоритетом;
- `-` — арифметическое отрицание (если операнд числовой), измненение `+` на `-` и наоборот (если операнд — строка, начинающаяся с `+` или `-`), приписывание знака `-` в остальных случаях;
- `~` — побитное отрицание;
- `+` — ничего не делает;
- `\` — получение ссылки.

Бинарные операторы

- `*` — умножение;
- `/` — деление;
- `%` — вычисление остатка от деления;
- `x` — повторение строки или списка;
- `+` — сложение;
- `-` — вычитание;
- `.` — конкатенация строк;
- `<<`, `>>` — побитные сдвиги.

Операторы сравнения

- `<`, `>`, `<=`, `>=`, `==`, `!=`, `<=>` — для чисел;
- `lt`, `gt`, `le`, `ge`, `eq`, `ne`, `cmp` — для строк.

Побитные операторы

- `&`, `|`, `^`, `.`

Логические операторы

- `&&` — конъюнкция;
- `and` — конъюнкция с низким приоритетом;
- `||` — дизъюнкция;
- `or` — дизъюнкция с низким приоритетом.

Оператор диапазона `..`
Тернарный условный оператор `?:`.
Операторы присваивания:

- `**=`, `+=`, `*=`, `&=`, `<<=`, `&&=`;
- `-=`, `/=`, `|=`, `>>=`, `||=`;
- `.=`, `%=`, `^=`;
- `x=`.

7.3. Управляющие структуры

Условные операторы

```
if ( condition ) {
    statements
} elsif ( other condition ) {
    statements
} else {
    statements
}

unless ( condition ) { # if ( !condition )
    statements
}

statement if condition;
statement unless condition;
```

Циклы

```
while ( condition ) {
    statements
}

until ( condition ) { # while ( !condition )
    statements
}

statement while condition;

statement until condition; # while !condition

for (initial-expression; condition; increment-expression) {
    statements
}

foreach my $val (@array) {
    statements involving $val
}

statement involving $_ foreach @array;
```

`next`, `last`, `redo` с опциональной меткой.

7.4. Подпрограммы

```
sub name {
    extract params from @_
    return value
}
```

Подпрограммы необходимо объявлять до использования либо заключать аргументы в скобки. Внутри программы аргументы доступны через массив `@_`. Изменение элементов массива изменяет фактические параметры.

7.5. Регулярные выражения

Метасимволы:

- `\` — особое значение следующего символа:
- `^` — начало строки;
- `.` — любой символ (кроме `\n`);
- `$` — конец строки;
- `|` — выбор;
- `()` — группировка;
- `[]` — класс символов.

Кванторы:

- `*` — 0 или более раз;
- `+` — 1 или более раз;
- `?` — 0 или 1 раз;
- `{n}` — ровно n раз;
- `{n,}` — n и более раз;
- `{n,m}` — от n до m раз.

Уменьшение жадности квантора: `?`.
Классы символов

- `\w` — алфавитно-цифровые символы и подчеркивание;
- `\W` — `[^\w]`;
- `\s` — пробельные символы;
- `\S` — `[^\s]`;
- `\d` — цифры;
- `\D` — `[^\d]`.

Совпавшие группы, заключенные в скобки, доступны через `$1`, `$2` и т.д.
Операторы, использующие регулярные выражения:

- `$s =~ m//`: поиск совпадения;
- `$s =~ s///`: замена одного выражения на другое.

Модификаторы:

- `i`: регистронезависимый поиск;
- `g`: поиск всех вхождений;
- `s`: `.` включает `\n`;
- `m`: `^` и `$` совпадают с началом/концом каждой строки внутри переменной.

7.6. Модуль CGI

Модуль CGI упрощает создание CGI-программ: разбирает параметры из `QUERY_STRING` и предоставляет функции для вывода HTML-кода.

```
use CGI qw/:standard/;
print header,
      start_html('hello world'),
      h1('hello world'),
      end_html;
```

Функция `param()` возвращает POST-параметры, `url_param()` — GET-параметры.

Для генерации элементов HTML существуют одноименные функции: `ul()`, `li()`, `a()`, `p()`, `blockquote()`, ...

8. PHP

Кроссплатформенный язык программирования, изначально предназначенный для web. Первая версия создана Rasmus Lerdorf, начиная с PHP 2 разработку возглавляют Zeev Suraski и Andi Gutmans.

- Первая версия PHP (виде набора Perl-скриптов) — 1994 г.
- Переписанная на C версия — 8 июня 1995 г. названием «Personal Home Page Tools».
- PHP 2 — ноябрь 1997 г., название изменено на «PHP: Hypertext Preprocessor».
- PHP 3 — июнь 1998 г.
- Zend Engine (переписанное ядро PHP) — 1999 г.
- PHP 4 — май 2000 г.
- PHP 5 — 13 июля 2004 г.

Ссылки:

- http://www.php.net/
- http://en.wikipedia.org/wiki/PHP

8.1. Синтаксис языка

```
virtually anything here
<?php
/*
  multiline comment
 */
statement; // single-line comment
?>
and again any trash here
```

Другие варианты PHP-тегов:

- `<script language="php"></script>`;
- `<? ?>`;
- `<?= expr ?>` — `<?php echo expr; ?>`;
- `<% %>`.

Подключение дополнительных файлов:

- `include`;
- `include_once`;
- `require`;
- `require_once`.

8.2. Переменные

- boolean (`false`, `true`);
- integer
- float (floating-point number, aka 'double')
- string
- array
- object
- resource
- null (`null`).

Проверка типа переменной:

- `gettype()` возвращает тип в виде строки;
- `is_bool()`;
- `is_int()`;
- `is_float()`;
- `is_string()`;
- `is_array()`;
- `is_object()`;
- `is_resource()`;
- `is_null()`.

8.3. Функции

```
function foo($arg_1, $arg_2, /* ..., */ $arg_n)
{
   echo "Example function.\n";
   return $retval;
}
```

Содержание