
Яндекс

Яндекс

Механизм работы браузера

Сергей Пузанков

Руководитель группы разработки поисковых интерфейсов

Основная задача браузера: запросить и отобразить данные

“How browsers work”

Tali Garsiel

http://clck.ru/lvu7

Движок отображения

- Парсинг HTML — построение DOM-дерева
- Парсинг CSS
- Построение render tree (дерева отображения)
- Reflow / Layout (Компоновка)
- Repaint (Отрисовка)

Парсинг HTML и CSS

Парсинг HTML и CSS

- браузеры всеядны
- последствия ошибок часто не очевидны
- валидатор это хорошо, но не панацея

Основа популярности HTML&CSS — прощающая обработка ошибок

Можно писать невалидный HTML — браузеры это поймут:

```
<p>Параграф не закрылся, но это нормально

<blah>
  Не описанный в спецификациях тег — тоже ок.
</blah>

<ul>
  <li>Элементы списка не обязаны закрываться
  <li>и иногда это может быть даже удобно:
    <ul>
      <li>Простой вложенный список
         </ul>
      <li>А это исходный список
    </ul>
</ul>
```

Валидатор — не «закон».
Но с валидным кодом
легче искать свои ошибки.

Парсинг HTML и CSS

- неверный HTML может заметно изменить структуру конечного DOM
- неверный CSS может или не отобразиться или «съесть» половину стилей

Ошибочный HTML может заметно изменить структуру конечного DOM

Было:

```
<table>
  <td>Ячейка таблицы</td>
  <div>Внезапно, див</div>
</table>
```

Стало:

```
<div>Внезапно, див</div>
<table>
  <tbody>
    <tr>
      <td>Ячейка таблицы</td>
    </tr>
  </tbody>
</table>
```

Неверный CSS не отобразится или «съест» половину стилей

```
.foo {
    width: 10em
    height: 10em;
    background: red;
}
.bar {
    width: 10bem;
    height: 10em;
    background: red;

.baz {
    width: 10em;
    height: 10em;
    background: red;
}
```

Неверный CSS не отобразится или «съест» половину стилей

```
.foo {
    width: 10em height: 10em;
    background: red;
}
.bar {
    width: 10bem;
    height: 10em;
    background: red;
    .baz { width: 10em; height: 10em; background: red;
}
```

Неверный CSS не отобразится или «съест» половину стилей

```
.foo {
    background: red;
}
.bar {
    height: 10em;
    background: red;
}
```

Порядок обработки

- Синхронность скриптов
- Параллельная загрузка остальных данных
- Синхронность таблиц стилей

Render tree

Render tree

Render tree может отличаться от DOM-дерева:

- не все блоки из DOM попадают в дерево отображения
- в некоторых случаях в render tree будет создаваться больше блоков, чем есть в DOM

Не все блоки из DOM попадают в дерево отображения

```
<head>
  <title>Шапка не отобразится</title>
</head>

<body>
  <noscript>
    Если JS работает – noscript не отобразится
  </noscript>
</body>

<div style="display: none;">
    Некоторые стили «выключают» блоки из дерева
    отображения
</div>
```

В некоторых случаях в render tree будет создаваться больше блоков, чем есть в DOM

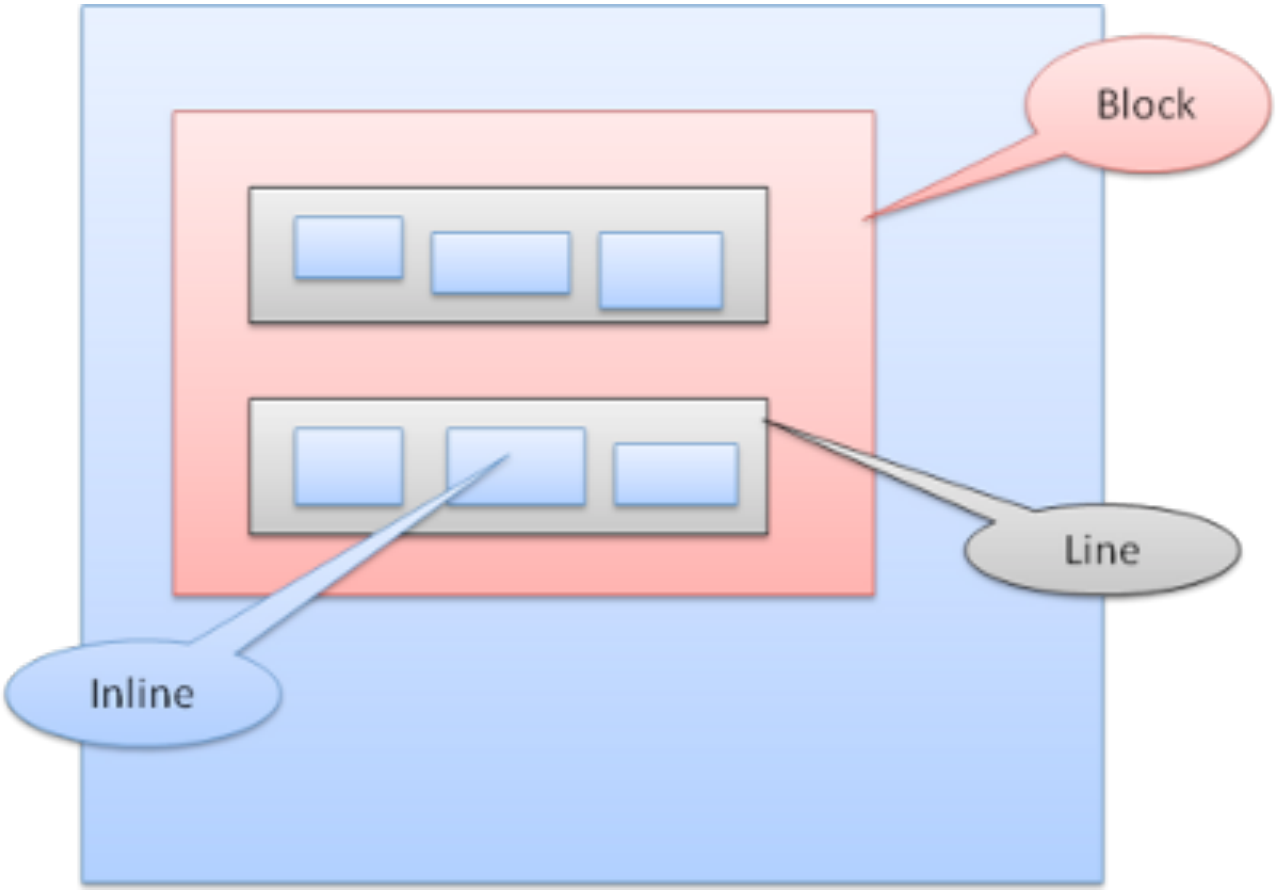
```
div:before {
  content: ”hello”;
}

div:after {
  content: ”world”;
}
```

В некоторых случаях в render tree будет создаваться больше блоков, чем есть в DOM

```
<div>
  Какой-то текстовый контент
  <div>Блочный элемент</div>
  Продолжение текстового контента
</div>
```

```
<div>
  <блок>Какой-то текстовый контент</блок>
  <div>Блочный элемент</div>
  <блок>Продолжение текстового контента</блок>
</div>
```

Инлайновые блоки



Применение стилей

Порядок применения стилей

- стили браузера
- стили пользователя
- стили автора
- важные стили автора
- важные стили пользователя

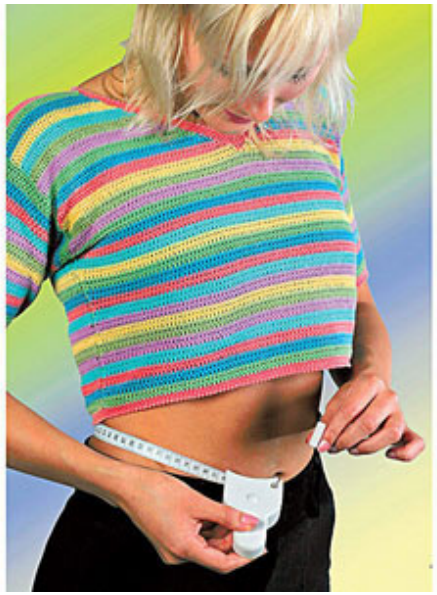
Порядок применения стилей

- browserDefault.css
- myDefaultStyles.css
- siteStyles.css
- siteStyles.css !important
- myDefaultStyles.css !important

Специфичность

*	0,0,0,0,0
div, :first-line	0,0,0,0,1
.some-class	0,0,0,1,0
#SomeID	0,0,1,0,0
style=""	0,1,0,0,0
div + !important	1,0,0,0,1
style="" + !important	1,1,0,0,0

Reflow



Reflow

- глобальный reflow
- инкрементный reflow

Глобальный Reflow

- первоначальное отображение страницы
- изменение размеров окна браузера
- изменение размеров шрифта в браузере
- и т.п.

Инкрементный Reflow

- вставка новых элементов в DOM
- изменение атрибутов (например, смена класса)
- смена состояния
  - наведение курсора на элемент
  - выбор чекбокса
  - фокус в поле текстового ввода
- и т.д.

Рассчёт ширины

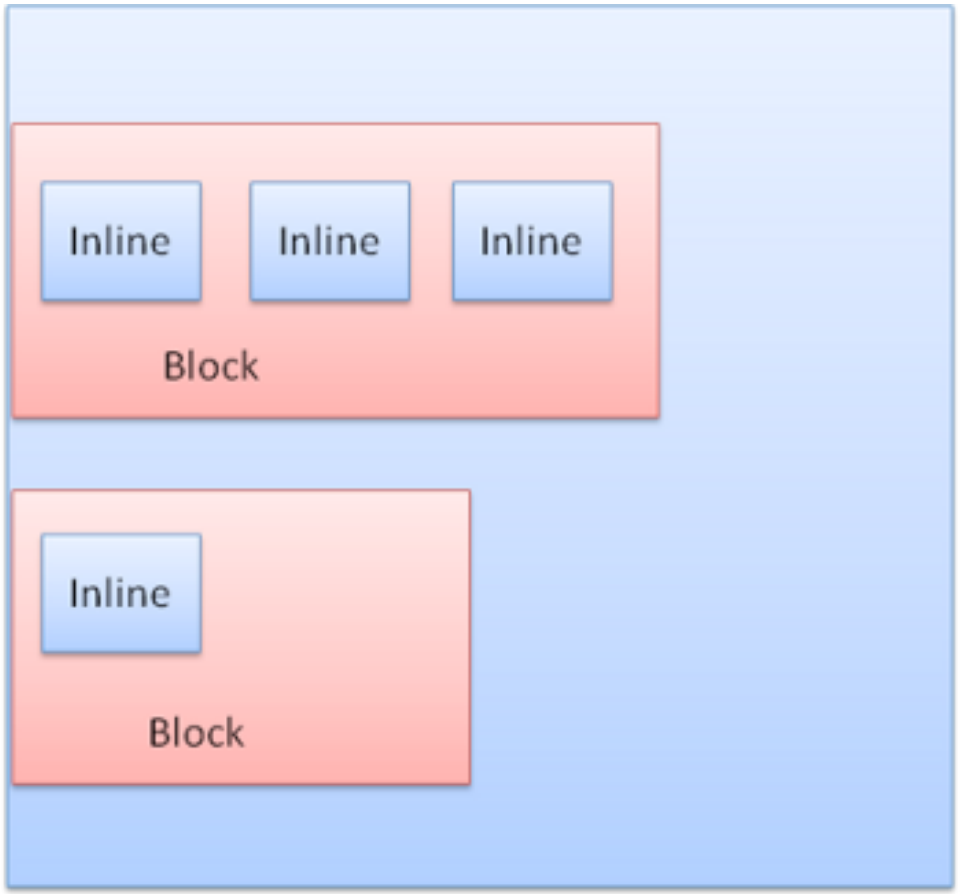
Ширина расчитывается на основе:

- ширины контейнера (родителя)
- свойства width в стилях текущего блока
- размеров полей и рамок (margin,padding,border)

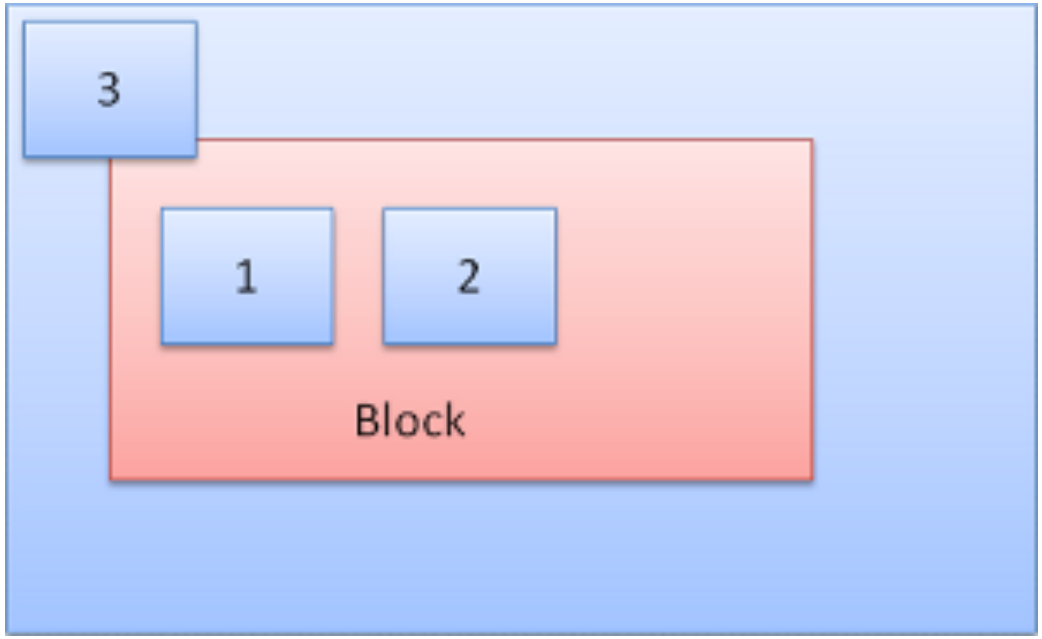
Потоки

- Стандартный поток
- Абсолютное позиционирование
- Стандартный поток с float элементами

Стандартный поток



Абсолютное позиционирование



Плавающие блоки

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.

Repaint

Порядок repaint

- цвет фона
- фоновое изображение
- граница
- тень
- вложенные элементы
- outline

Стековый контекст

`z-index` и всё такое

Подробнее:

http://clck.ru/8aCxu

Нужно чтобы:

- области refrow и repaint были как можно меньше
- пересчет происходил как можно реже

Ссылки

- How browsers work

  http://clck.ru/Ivu7
- Как работают браузеры

  http://clck.ru/8aCyc
- Статья про позиционирование и стековый контекст

  http://clck.ru/8aCxu
- How Browsers Lay Out Web Pages

  http://clck.ru/8aCzE

Яндекс

Сергей Пузанков

Руководитель группы
разработки
поисковых интерфейсов

puzankov@yandex-team.ru

@puzankovcom

Спасибо!