



Как устроен современный браузер?

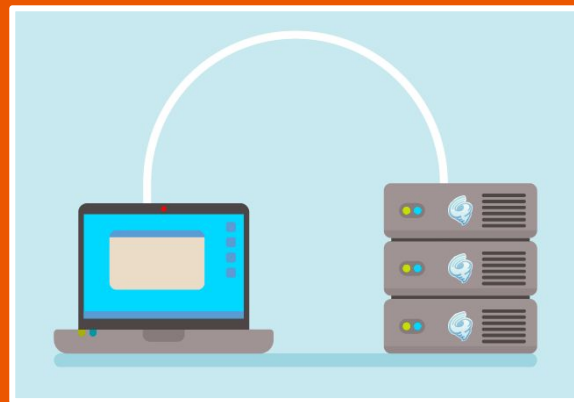
Презентацию подготовила студентка 1 курса ИВТ, 3 подгруппа
Елкина Галина


Тали Гарсиэль начала карьеру в 2000 году, когда познакомилась с "неудобной" уровневой моделью Netscape.

Она также опубликовала краткое руководство по обработке кода на стороне клиента.



Основное предназначение браузера - отображать веб- ресурсы



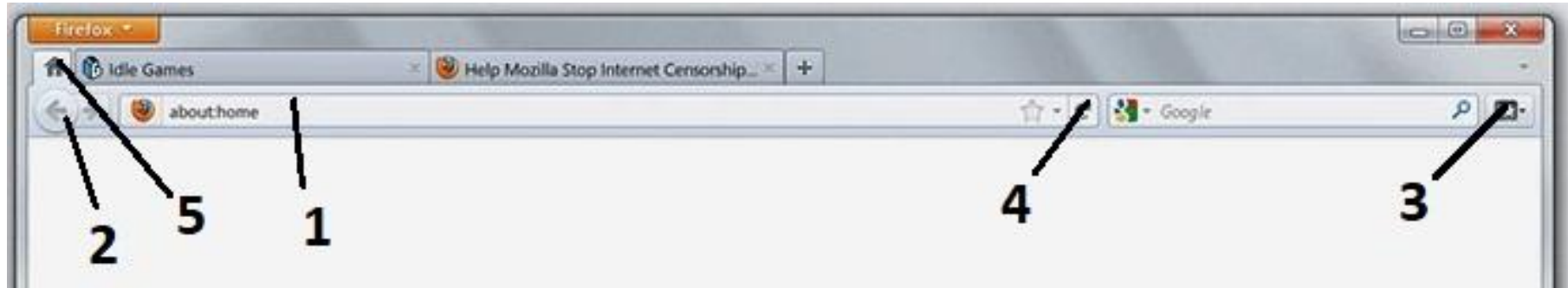


Основные компоненты браузера



Пользовательский интерфейс

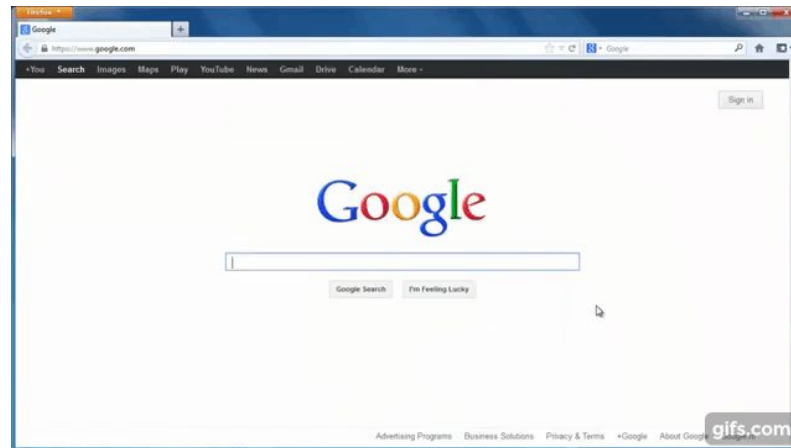
1. Адресная строка для ввода URI
2. Кнопки навигации "Назад" и "Вперед"
3. Закладки
4. Кнопки обновления и остановки загрузки страницы
5. Кнопка "Домой" для перехода на главную страницу

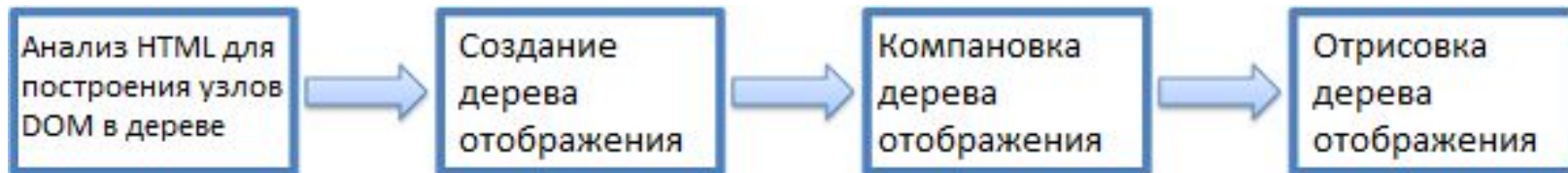


Модуль отображения

Отвечает за вывод запрошенного содержания на экране
браузера

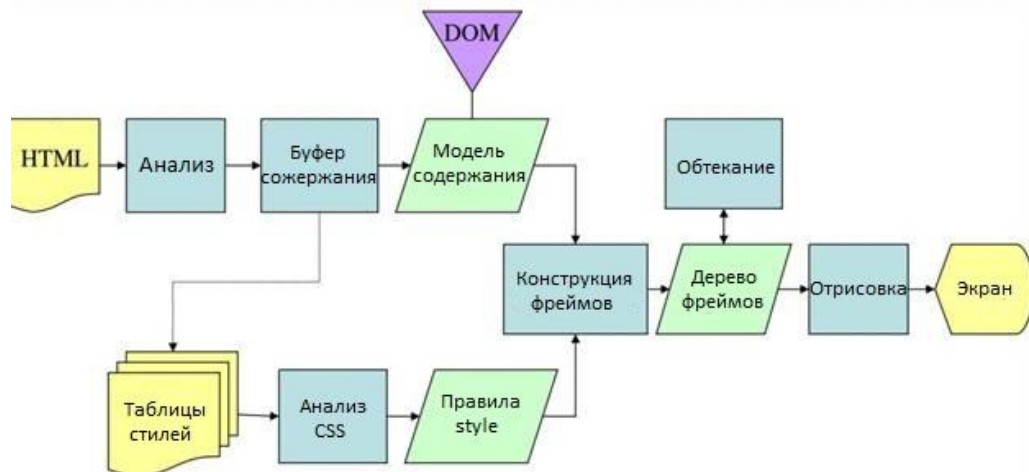
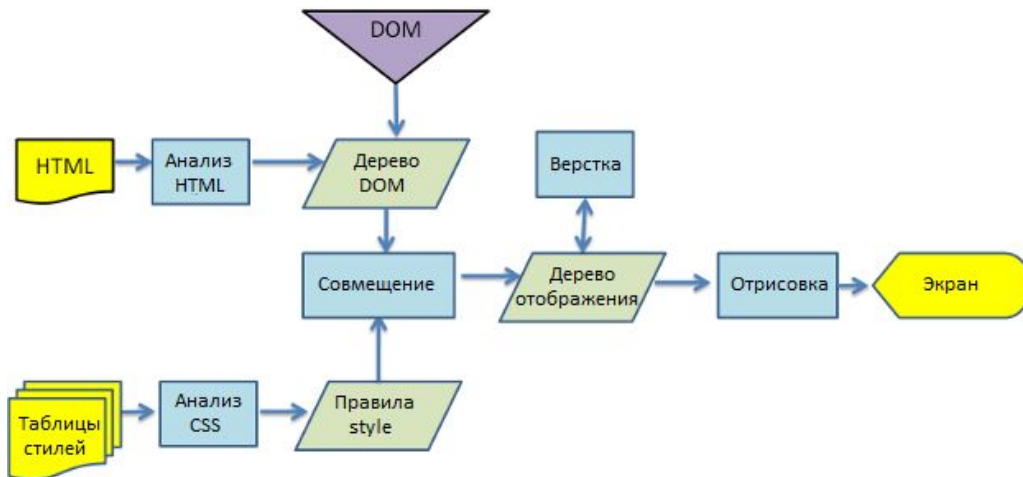
В Firefox применяется Gecko –
собственная разработка
Mozilla, а в Safari и Chrome
используется WebKit





Важно понимать, что это последовательный процесс. Модуль отображения старается вывести содержание на экран как можно скорее. Поэтому одни части документа анализируются и выводятся на экран, в то время как другие только передаются по сети.

WebKit



Gecko

правила

синтаксическое
дерево

нисходящий
анализатор

токен

формат

**Синтаксический
анализ**

словарь

дерево
узлов

**Лексический
анализ**

восходящий
анализатор

синтаксис

бесконтекстная
грамматика

язык

лексема

Генераторы

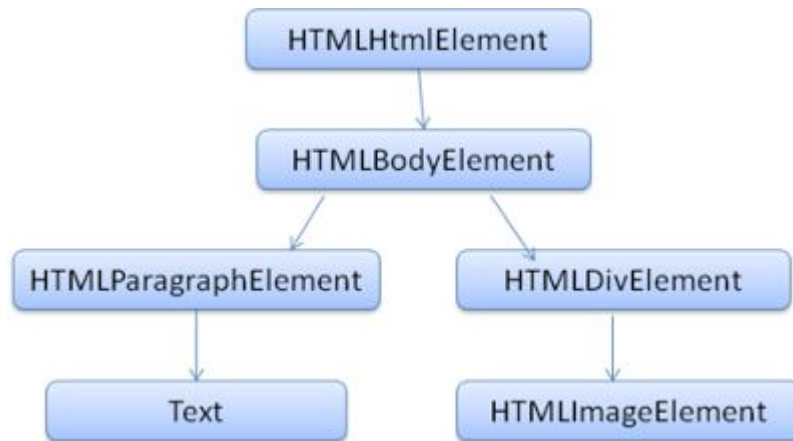
Flex и Bison

Интересно, что!

Ни один из стандартных анализаторов не подходит для языка HTML. Для определения HTML существует формальный стандарт – формат DTD (Document Type Definition)

DOM (Document Object Model)

```
<html>
  <body>
    <p>
      Hello World
    </p>
    <div> 
    </div>
  </body>
</html>
```



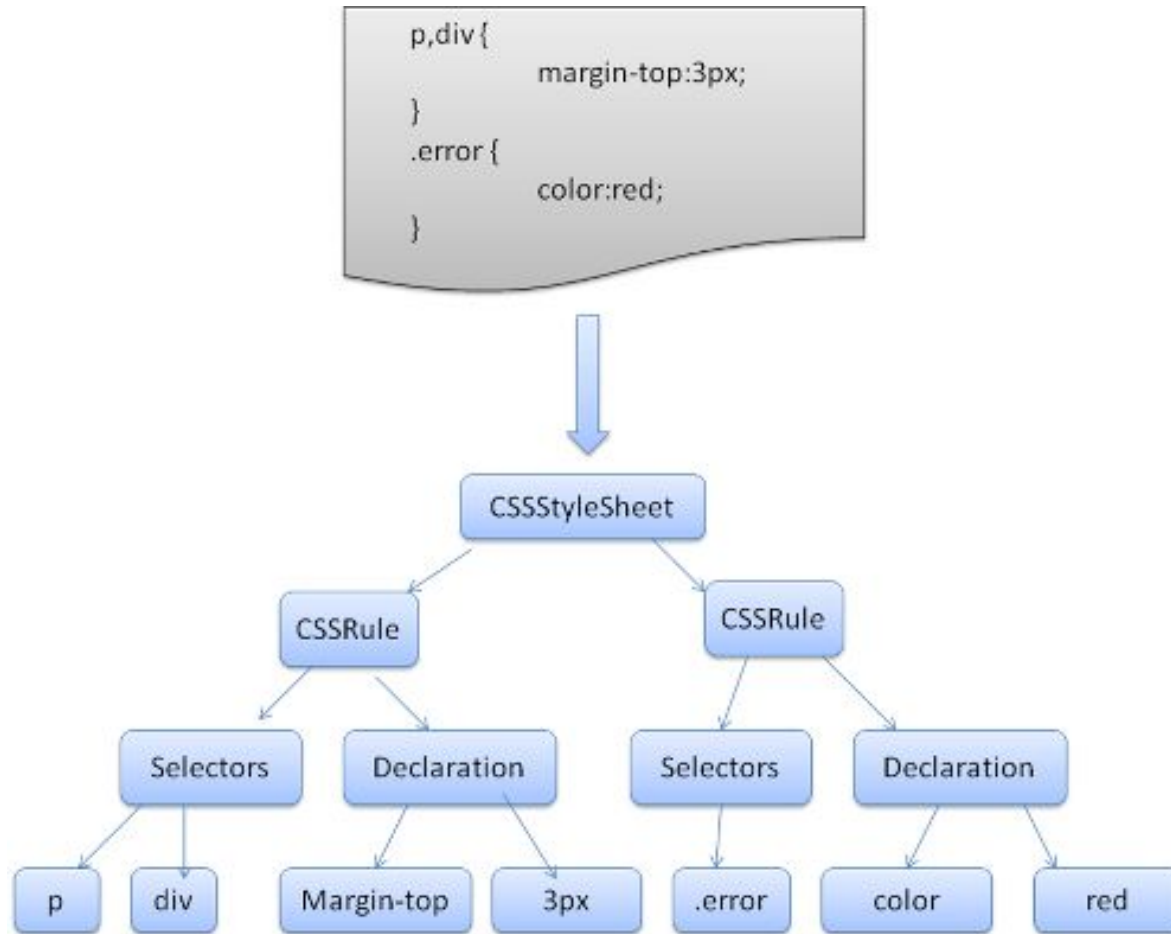
Обработка ошибок браузерами

Так как многие документы HTML содержат ошибки, синтаксический анализатор должен уметь обрабатывать как минимум перечисленные ниже типы ошибок.

1. Использование добавляемого элемента явно запрещено одним из внешних тегов. В этом случае необходимо закрыть все теги, кроме того, который запрещает использование данного элемента, и добавить этот элемент в самом конце.
2. Элемент нельзя добавить напрямую. Возможно, автор документа забыл вставить тег между элементами (или такой тег необязателен). Это касается тегов HTML, HEAD, BODY, TBODY, TR, TD, LI.
3. Блочный элемент добавлен внутрь строчного. Необходимо закрыть все строчные элементы вплоть до следующего в иерархии блочного элемента.

Если это не помогает, необходимо закрывать элементы, пока не появится

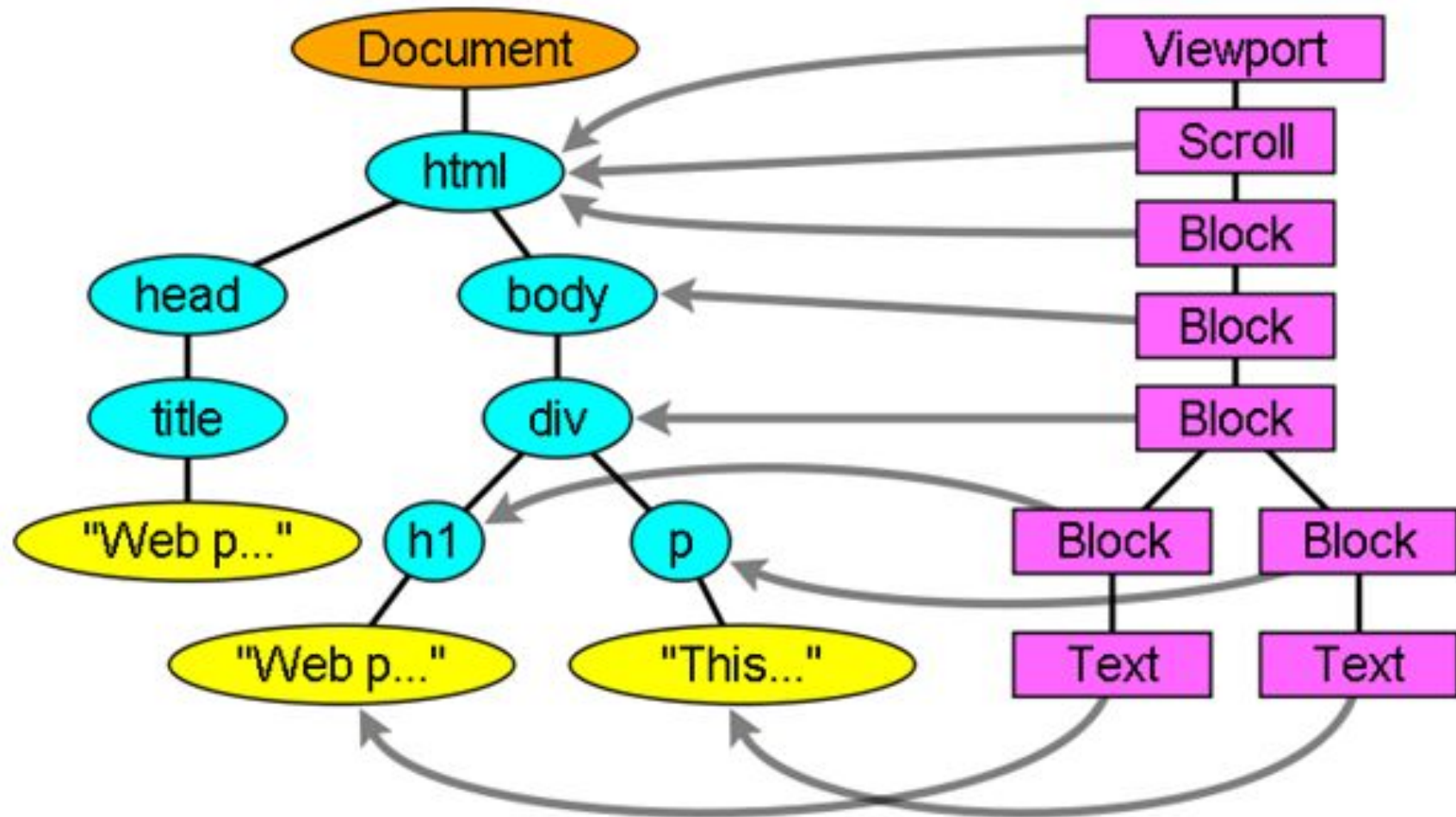
Синтаксический анализ CSS





Порядок обработки скриптов и таблиц стилей


Веб-документы придерживаются синхронной модели. Синтаксический анализ документа откладывается до завершения выполнения скрипта. Разработчик мог пометить скрипт тегом `defer`, чтобы синтаксический анализ документа можно было выполнять до завершения выполнения скрипта. В HTML5 появилась возможность пометить скрипт как асинхронный (`asynchronous`), чтобы он анализировался и выполнялся в другом потоке. Ориентировочный синтаксический анализ является механизмом оптимизации. При выполнении скриптов остальные части документа анализируются в другом потоке. Обратите внимание: ориентировочный анализатор не изменяет дерево DOM. Так как таблицы стилей не вносят изменений в дерево DOM, теоретически останавливать анализ документа, чтобы дождаться их обработки, бессмысленно.





с **ВЫЧИСЛЕНИЕМ СТИЛЕЙ** связан ряд сложностей

1. Данные стилей содержат множество свойств и бывают очень объемны, что может вести к проблемам с памятью.
2. Поиск подходящих правил для каждого элемента может замедлить работу, если код не оптимизирован.
3. Применение правил подразумевает определение иерархии для достаточно сложных перекрывающихся правил.



Правила стилей извлекаются из нескольких источников:

- Правила CSS, определенные во внешних таблицах стилей или в элементах `style: p {color:blue}`
- Атрибуты строчных объектов `style: <p style="color:blue" />`
- Визуальные атрибуты HTML (сопоставляются с подходящими правилами стилей): `<p bgcolor="blue" />`

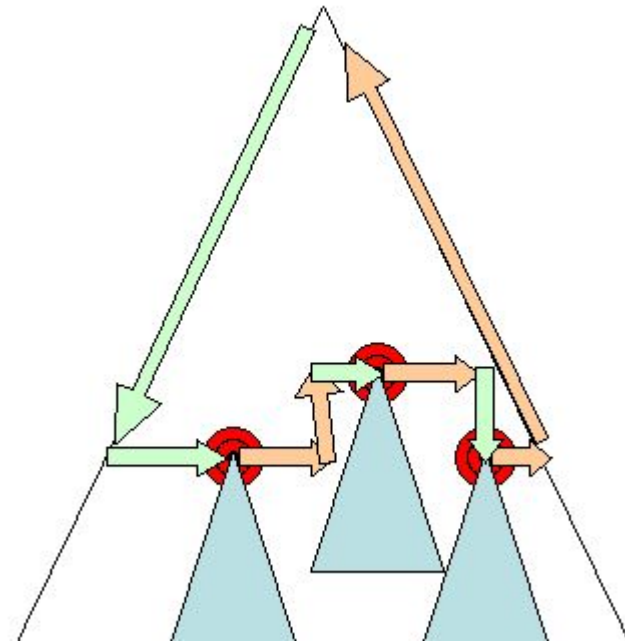
Порядок приоритета таблиц стилей

1. Объявления браузера
2. Обычные объявления пользователя
3. Обычные объявления автора
4. Важные объявления автора
5. Важные объявления пользователя (!important)


Когда только что созданный объект отображения включается в дерево, он не имеет ни размера, ни положения. Расчет этих значений называется **компоновкой**.

Чтобы не выполнять перекомпоновку при каждом изменении, браузеры помечают измененный объект отображения и его дочерние элементы как **"грязные"**, то есть требующие перекомпоновки.

Если компоновка выполняется для всего дерева отображения, она называется **глобальной**. При **инкрементной** компоновке изменяются только "грязные" объекты отображения.



Инкрементная компоновка, при которой обрабатываются только "грязные" объекты отображения и их дочерние элементы



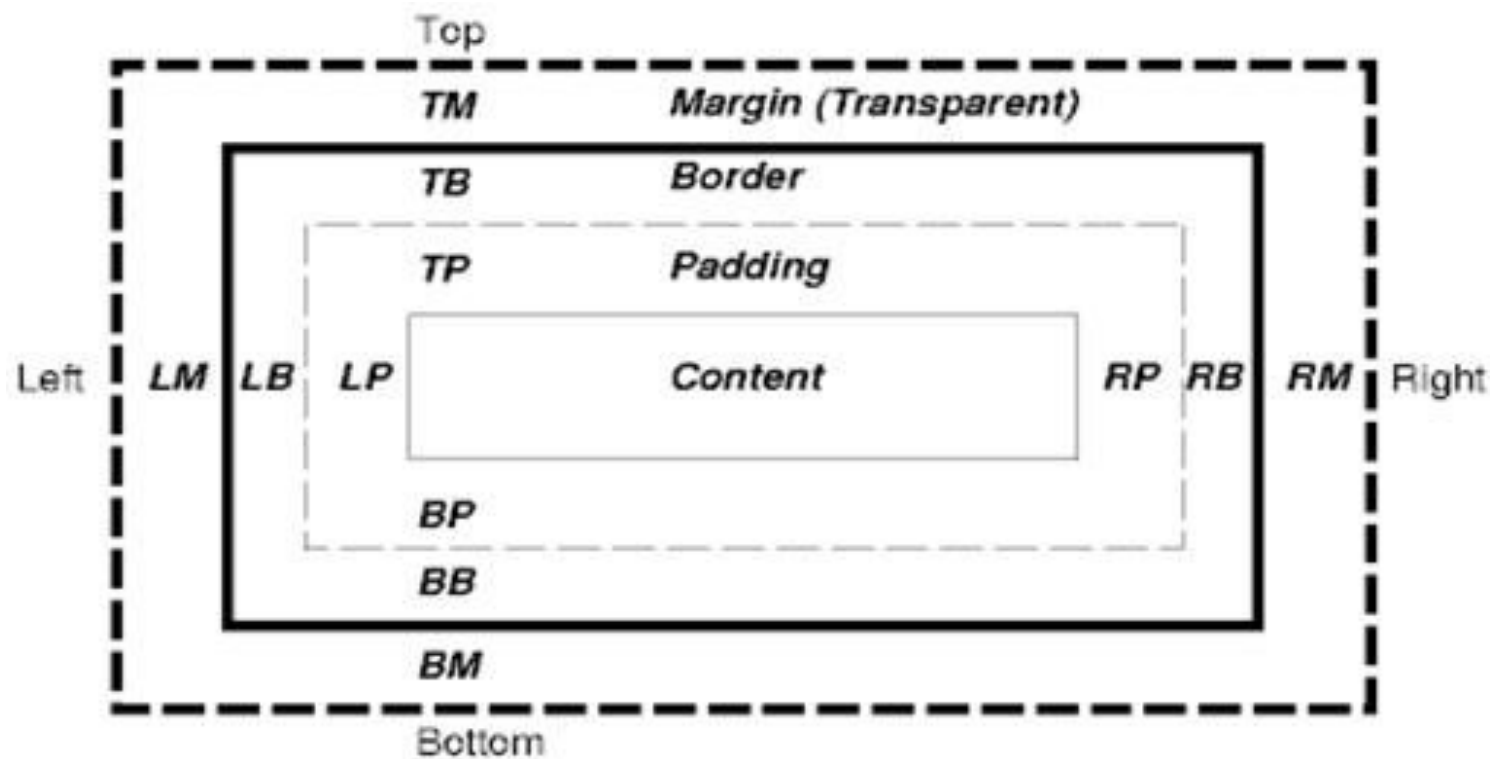
На этапе **отрисовки** для каждого объекта отображения по очереди вызывается метод paint и их содержание выводится на экран. При **глобальной** отрисовке все дерево отрисовывается целиком, а при **инкрементной** – только отдельные объекты отображения

Порядок добавления блочных объектов в стек таков:

1. Цвет фона
2. Фоновое изображение
3. Рамка
4. Дочерние объекты
5. Внешние границы

Основной поток браузера представляет собой цикл событий – бесконечный цикл, который поддерживает рабочие процессы.

Визуальная модель CSS2



- Margin edge
- Border edge
- - - Padding edge
- Content edge

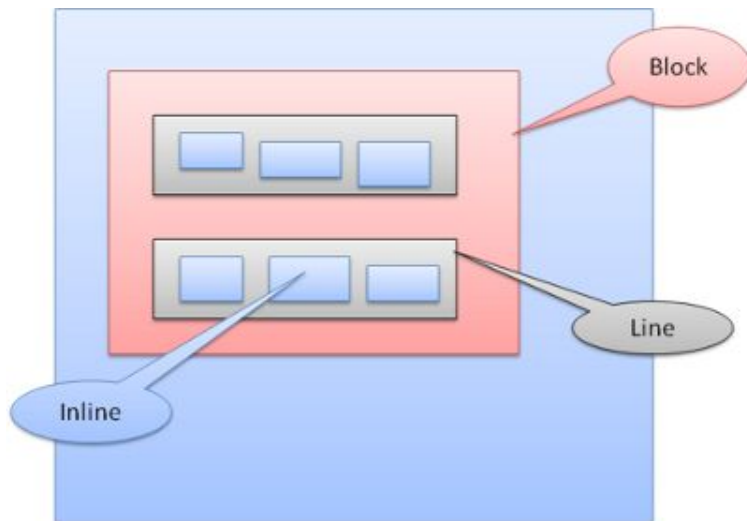
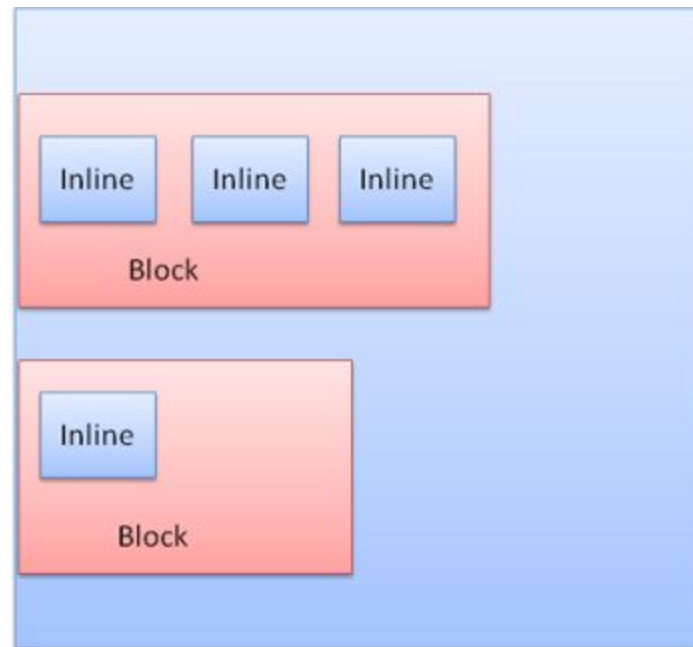
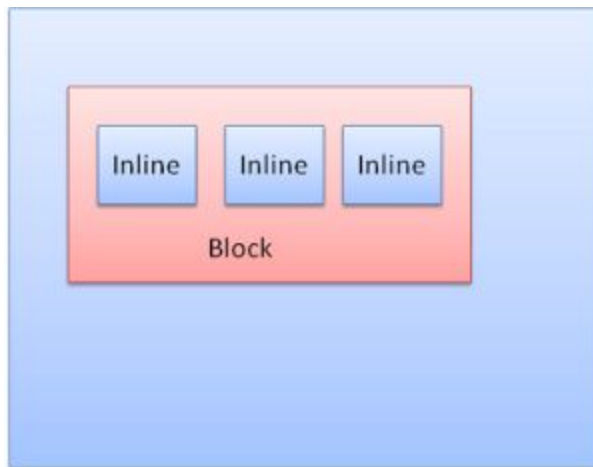
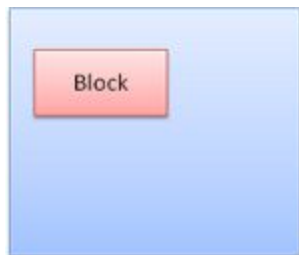


Существует три схемы позиционирования.

1. **Стандартная**: объект размещается в документе согласно своему положению в дереве отображения и в дереве DOM, а также своему типу и размерам окна.
2. **Плавающая**: объект сначала компоуется по стандартной схеме, затем смещается в крайнее правое или крайнее левое положение.
3. **Абсолютная**: положение объекта в дереве отображения отличается от его положения в дереве DOM.

Схема позиционирования определяется свойством `position` и атрибутом `float`.

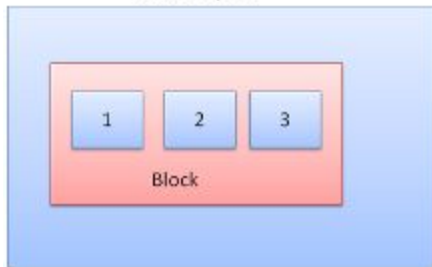
- Значения `static` и `relative` соответствуют стандартной схеме.
- Значения `absolute` и `fixed` соответствуют абсолютной схеме.



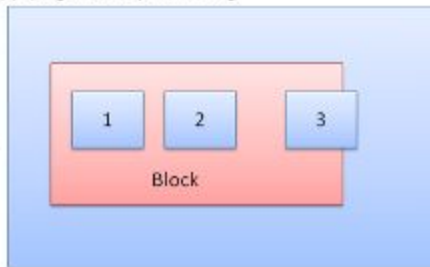
Типы окон

```
<html>
  <div>
    <span>1</span>
    <span>2</span>
    <span style="position:relative;left:5px">3</span>
  </div>
</html>
```

Normal layout



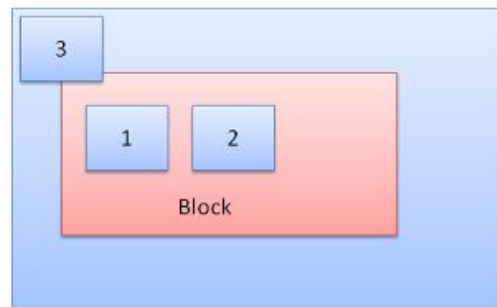
Applying relative positioning



Относительное позиционирование

Фиксированное позиционирование

```
<html>
  <div>
    <span>1</span>
    <span>2</span>
    <span style="position:fixed;top:5px;left:5px">3</span>
  </div>
</html>
```



Спасибо за внимание!

