极客大学前端进阶训练营

程劭非 （winter）

前手机淘宝前端负责人

浏览器工作原理

css计算

浏览器


URL
HTTP
HTML
parse
DOM
css
computing
DOM with
CSS
layout
DOM with
position
render
Bitmap

环境准备

npm install css

第一步 收集CSS规则

第一步

- 遇到style标签时，我们把CSS规则保存起来
- 这里我们调用CSS Parser来分析CSS规则
- 这里我们必须要仔细研究此库分析CSS规则的格式

第二步 添加调用

第二步总结

- 当我们创建一个元素后，立即计算CSS
- 理论上，当我们分析一个元素时，所有CSS规则已经收集完毕
- 在真实浏览器中，可能遇到写在body的style标签，需要重新CSS计算的情况，这里我们忽略

第三步 获取父元素序列

第三步总结

- 在computeCSS函数中，我们必须知道元素的所有父元素才能判断元素与规则是否匹配
- 我们从上一步骤的stack，可以获取本元素所有的父元素
- 因为我们首先获取的是“当前元素”，所以我们获得和计算父元素匹配的顺序是从内向外

第四步 拆分选择器

第四步总结

- 选择器也要从当前元素向外排列
- 复杂选择器拆成针对单个元素的选择器，用循环匹配父元素队列

第五步 计算选择器与元素匹配

第五步总结

- 根据选择器的类型和元素属性，计算是否与当前元素匹配
- 这里仅仅实现了三种基本选择器，实际的浏览器中要处理复合选择器
- 作业（可选）：实现复合选择器，实现支持空格的Class选择器

第六步 生成computed属性

第六步总结

- 一旦选择匹配，就应用选择器到元素上，形成computedStyle

第七步 确定规则覆盖关系

第七步总结

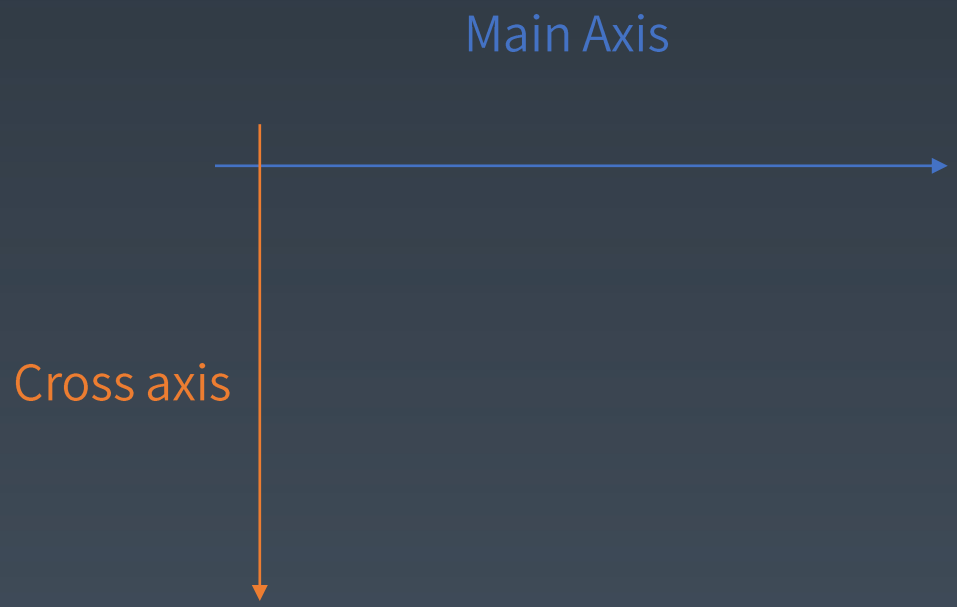
- CSS规则根据specificity和后来优先规则覆盖
- specificity是个四元组，越左边权重越高
- 一个CSS规则的specificity根据包含的简单选择器相加而成

浏览器工作原理

排版

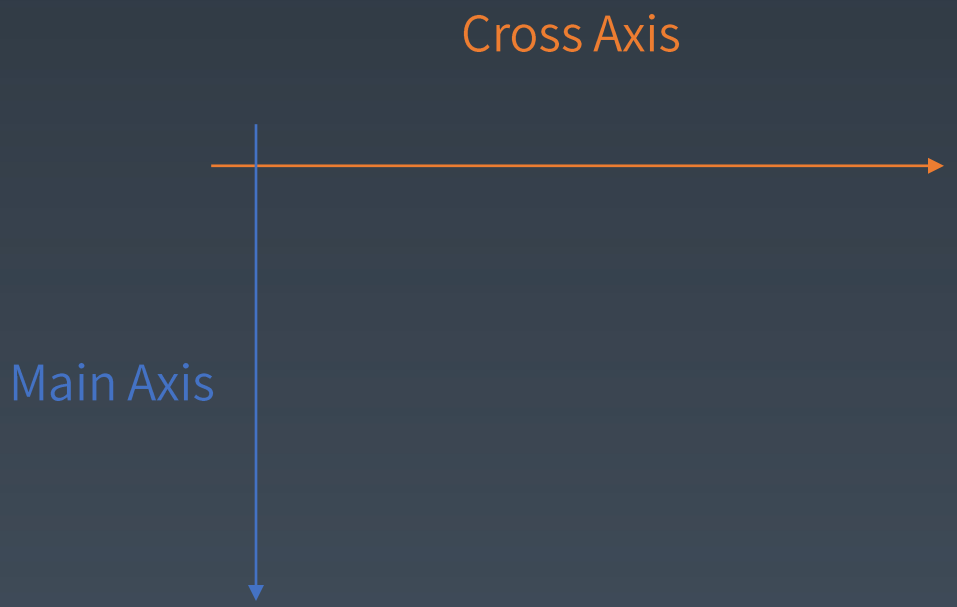
浏览器


URL
HTTP
HTML
parse
DOM
css
computing
DOM with
CSS
layout
DOM with
position
render
Bitmap



flex-direction:row

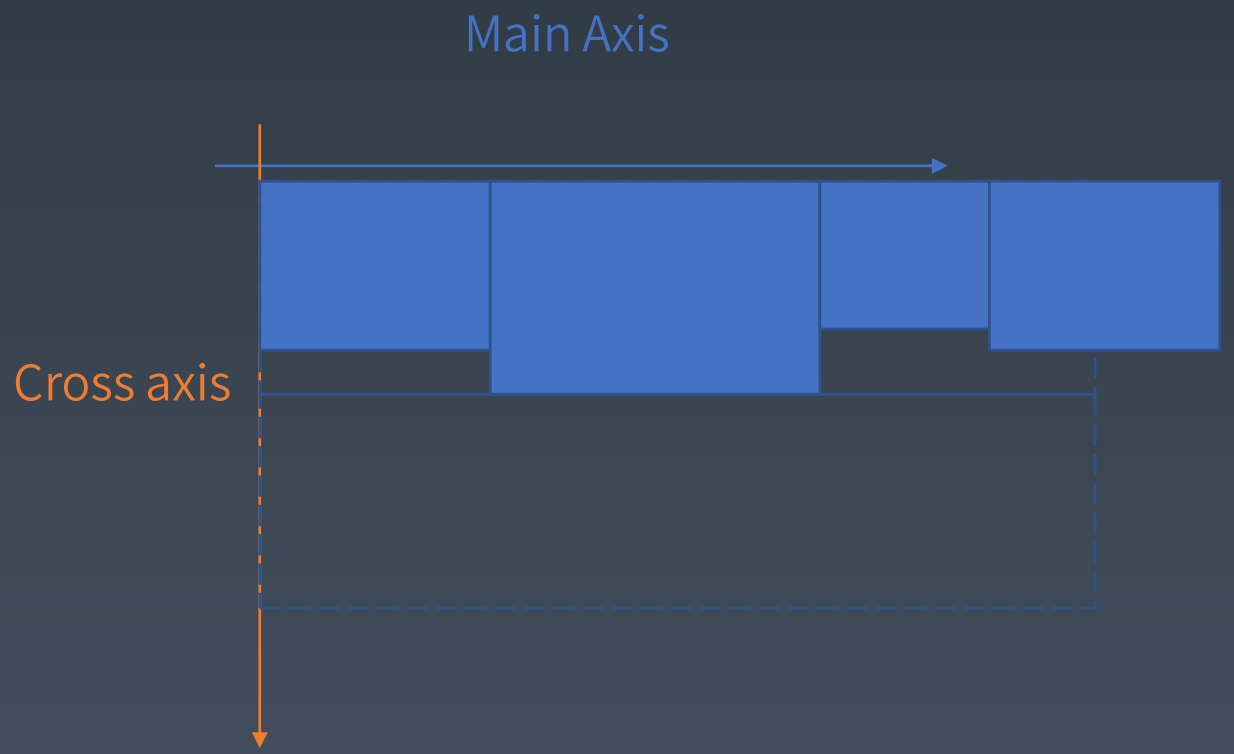
Main: width x left right
Cross: height y top
bottom



flex-direction:column

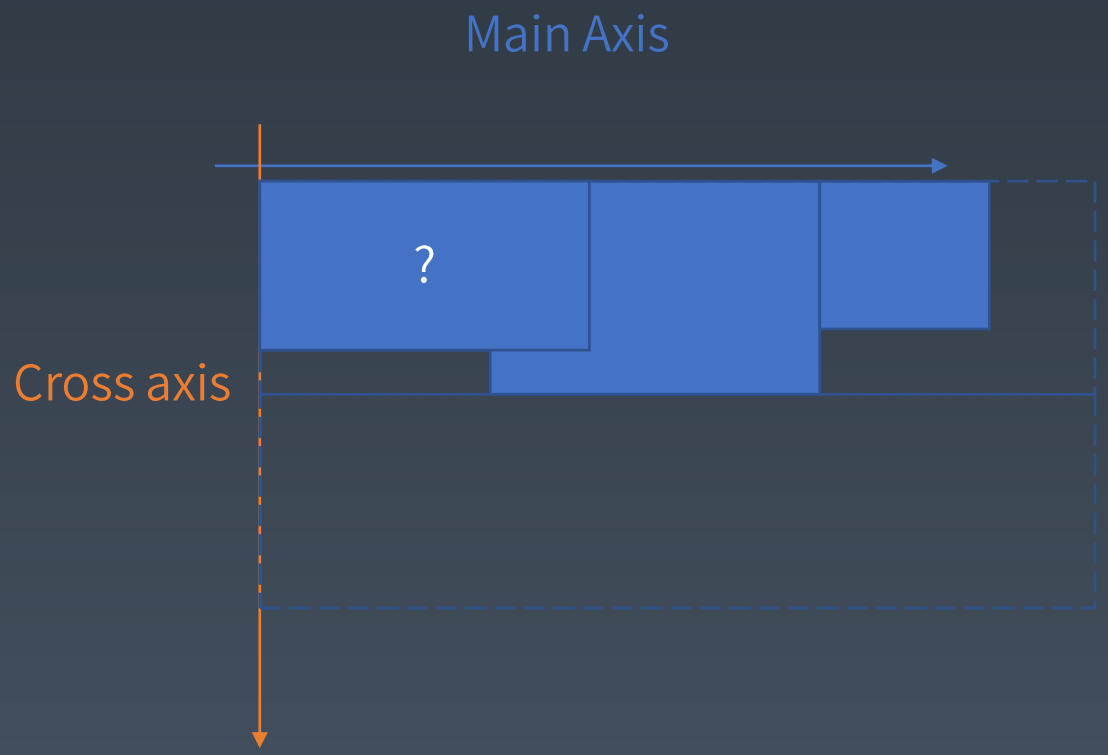
Main: height y top
bottom
Cross: width x left right

第二步 收集元素进行



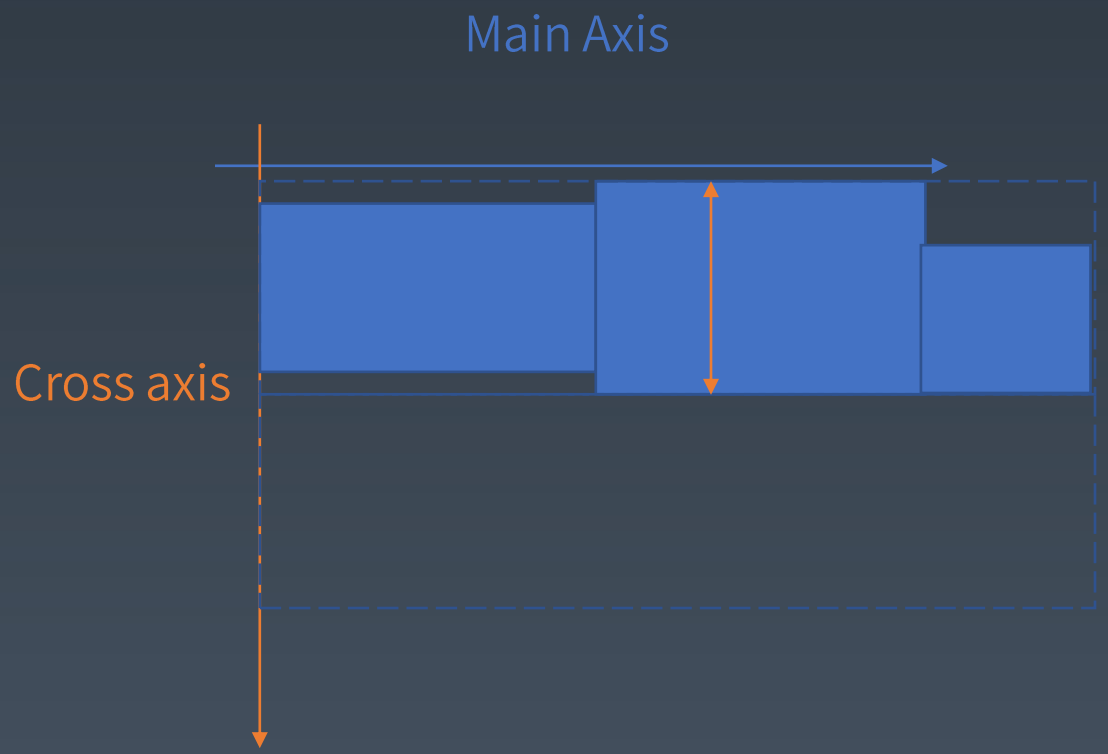
- 分行
  - 根据主轴尺寸，把元素分进行
  - 若设置了no-wrap，则强行分配进第一行

第三步 计算主轴



- 计算主轴方向
  - 找出所有Flex元素
  - 把主轴方向的剩余尺寸按比例分配给这些元素
  - 若剩余空间为负数，所有flex元素为0，等比压缩剩余元素

第四步 计算交叉轴



- 计算交叉轴方向
  - 根据每一行中最大元素尺寸计算行高
  - 根据行高flex-align和item-align，确定元素具体位置

总结


URL
HTTP
HTML
parse
DOM
css
computing
DOM with
CSS
layout
DOM with
position
render
Bitmap

浏览器工作原理

绘制

浏览器


URL
HTTP
HTML
parse
DOM
css
computing
DOM with
CSS
layout
DOM with
position
render
Bitmap

第一步 绘制单个元素

第一步 总结

- 绘制需要依赖一个图形环境
- 我们这里采用了npm包images
- 绘制在一个viewport上进行
- 与绘制相关的属性：background-color、border、background-image等

第二步 绘制DOM

第二步 总结

- 递归调用子元素的绘制方法完成DOM树的绘制
- 忽略一些不需要绘制的节点
- 实际浏览器中，文字绘制是难点，需要依赖字体库，我们这里忽略
- 实际浏览器中，还会对一些图层做compositing，我们这里也忽略了

THANKS!