# Bootstrap响应式布局原理

Bootstrap响应式布局是利用其栅格系统，对于不同的屏幕采用不同的类属性。在开发中可以只写一套代码在手机平板，PC端都能使用，而不用针对不同的设备分别写不同的代码。

首先栅格系统用于通过一系列的行（row）与列（column）的组合来创建页面布局。bootstrap根据不同屏幕尺寸，选择不同的栅格选项。栅格系统分为12列，即每行最多可容纳12列。若里，一个row内包含的列（column）大于12个（即，一行中的栅格单元超过12个单元），则会自动排版，总之，一行只能最多12列，如果一行（row）中包含了的列（column）大于12，多余的列所在的元素将作为一个整体另起一行排列。

而在Bootstrap中主要把屏幕分成了四种（这里以行（row）来说明）：

1) col-xs-\* 超小屏幕，手机 (宽度<768px)

2) col-sm-\* 小屏幕，平板 (宽度≥768px)

3) col-md-\* 中等屏幕，桌面显示器 (宽度≥992px)

4) col-lg-\* 大屏幕，桌面显示器 (宽度≥1200px)

# XML是什么，它可以做什么？

XML即可扩展标记语言，简单说，XML就是一种数据的描述语言，虽然它是语言，但是通常情况下，它并不具备常见语言的基本功能——被计算机识别并运行。只有依靠另一种语言，来解释它，使它达到你想要的效果或被计算机所接受。

XML应用面主要分为两种类型，文档型和数据型。下面介绍一下几种常见的XML应用：

1、自定义XML+XSLT=>HTML，最常见的文档型应用之一。XML存放整个文档的XML数据，然后XSLT将XML转换、解析，结合XSLT中的HTML标签，最终成为HTML，显示在浏览器上。

2、作为信息传递的载体。为什么说是载体呢？因为这些应用虽然还是以XML为基本形态，但是都已经发展出具有特定意义的格式形态。最典型的就是WEB SERVICE。

3、应用程序的配置信息数据。最典型的就是J2EE配置WEB服务器时用的web.XML。这个应用估计是很容易理解的了。我们只要将需要的数据存入XML，然后在我们的应用程序运行载入，根据不同的数据，做相应的操作。这里其实和应用2，有点类似，所不同的在于，数据库中的数据变化是个常态，而配置信息往往是较为静态，缺少变化的。

4、保存数据间的映射关系。如Hibernate。

# Ajax的优势是什么？简要介绍一下ajax核心javascript对象？

答案：2个方面

Ajax应用程序的优势：

1. 通过异步模式，提升了用户体验

2. 优化了浏览器和服务器之间的传输，减少不必要的数据往返，减少了带宽占用

3. Ajax引擎在客户端运行，承担了一部分本来由服务器承担的工作，从而减少了大用户量下的服务器负载。

2、AJAX的特点。

Ajax可以实现动态不刷新（局部刷新）

就是能在不更新整个页面的前提下维护数据。这使得Web应用程序更为迅捷地回应用户动作，并避免了在网络上发送那些没有改变过的信息。增强用户体验。

3.Ajax的核心是JavaScript对象XmlHttpRequest。该对象在Internet Explorer 5中首次引入，它是一种支持异步请求的技术。XmlHttpRequest可以使用JavaScript向服务器提出请求并处理响应，而不阻塞用户。通过XMLHttpRequest对象，Web开发人员可以在页面加载以后进行页面的局部更新。

# 说明redis雪崩的原因,以及如何防止血崩的发生。

什么是缓存雪崩？

当缓存服务器重启或者大量缓存集中在某一个时间段失效，这样在失效的时候，也会给后端系统(比如DB)带来很大压力。

通俗说法：就是缓存数据大规模同时失效，数据全部一次性去数据库查询，给数据库造成了巨大压力。

如何解决？

第一：

1：在缓存失效后，通过加锁或者队列来控制读数据库写缓存的线程数量。比如对某个key只允许一个线程查询数据和写缓存，其他线程等待。

2：不同的key，设置不同的过期时间，让缓存失效的时间点尽量均匀。

3：做二级缓存，A1为原始缓存，A2为拷贝缓存，A1失效时，可以访问A2，A1缓存失效时间设置为短期，A2设置为长期（此点为补充）

第二：

建立分布式缓存系统

实现多个缓存副本之间的一致性。为了保证系统的高可用性，缓存系统背后往往会接两套存储系统（如memcache，redis等），并且数据过期时间均匀分散开来。实现数据的分批过期。

# 谈谈js的this对象的理解

this是js的一个关键字，随着函数使用场合不同，this的值会发生变化。

但是有一个总原则，那就是this指的是调用函数的那个对象。

this一般情况下：是全局对象Global。作为方法调用，那么this就是指这个对象

# DOM怎样添加、移动、复制、创建和查找节点

// 创建新节点

createDocumentFragment() //创建一个DOM片段

createElement() //创建一个具体的元素

createTextNode() //创建一个文本节点

// 添加、移除、替换、插入

appendChild()

removeChild()

replaceChild()

insertBefore() //在已有的子节点前插入一个新的子节点

// 查找

getElementsByTagName() //通过标签名称

getElementsByName() //通过元素的Name属性的值(IE容错能力较强，会得到一个数组，其中包括id等于name值的)

getElementById() //通过元素Id，唯一性

# html5的新特性

(1) 绘画canvas

(2) 用于媒介回放的video和audio元素

(3) 本地离线存储localStorage长期存储数据，浏览器关闭后数据不丢失

(4) sessionStorage的数据在浏览器关闭后自动删除

(5) 语义化更好的内容元素，比如article,footer,header,nav,section

(6) 表单控件，calendar,date,time,email,url等

(7) 新的技术webworker,websockt,Geolocation

(8) 移出的元素

a.纯表现的元素：basefont,big,center,font等

b.产生负面影响的元素：frame frameset等

(9) ie8 7 6 支持通过document.createElemet 方法产生新的标签，可以利用这 一特性让这些浏览器支持html5新标签

# Html5的页面结构与html4或者更前的html页面结构有什么不同？

一个典型的WEB页面包含头部，脚部，导航，中心区域，侧边栏。现在如果我们想在在HTML4的HTML区域中呈现这些内容，我们可能要使用DIV标签，但是在HTML5中通过为这些区域创建元素名称使他们更加清晰，也使得你的HTML更加可读。

以下是形成页面结构的HTML5元素的更多细节：

• <header>：代表HTML的头部数据

• <footer>：页面的脚部区域

• <nav>：页面导航元素

• <article>：自包含的内容

• <section>：使用内部article去定义区域或者把分组内容放到区域里

• <aside>：代表页面的侧边栏内容

# CSS是什么?

CSS 通常称为CSS样式或层叠样式表，主要用于设置HTML页面中的文本内容（字体、大小、对其方式等）、图片的外形（高宽、边框样式、边距等）以及版面的布局等外观显示样式。

CSS可以是HTML页面更好看，CSS色系的搭配可以让用户更舒服，CSS+DIV布局更佳灵活，更容易绘制出用户需要的结构 (美化页面)。

# CSS选择器有哪些？

• id选择器 #id

• 类选择器 .class

• 标签选择器 div, h1, p

• 相邻选择器 h1 + p

• 子选择器 ul > li

• 后代选择器 li a

• 通配符选择器 \*

• 属性选择器 a[rel='external']

• 伪类选择器 a:hover, li:nth-child

# 说说Servlet 生命周期、初始化时机

Servlet 生命周期：

Servlet 加载—>实例化—>服务—>销毁。

init（）：

在Servlet的生命周期中，仅执行一次init()方法。它是在服务器装入Servlet时执行的，负责初始化Servlet对象。可以配置服务器，以在启动服务器或客户机首次访问Servlet时装入Servlet。无论有多少客户机访问Servlet，都不会重复执行init（）。

service（）：

它是Servlet的核心，负责响应客户的请求。每当一个客户请求一个HttpServlet对象，该对象的Service()方法就要调用，而且传递给这个方法一个“请求”（ServletRequest）对象和一个“响应”（ServletResponse）对象作为参数。在HttpServlet中已存在Service()方法。默认的服务功能是调用与HTTP请求的方法相应的do功能。

destroy（）：

仅执行一次，在服务器端停止且卸载Servlet时执行该方法。当Servlet对象退出生命周期时，负责释放占用的资源。一个Servlet在运行service()方法时可能会产生其他的线程，因此需要确认在调用destroy()方法时，这些线程已经终止或完成。

创建Servlet对象的时机：

1. Servlet容器启动时：读取web.xml配置文件中的信息，构造指定的Servlet对象，创建ServletConfig对象，同时将ServletConfig对象作为参数来调用Servlet对象的init方法。

2. 在Servlet容器启动后：客户首次向Servlet发出请求，Servlet容器会判断内存中是否存在指定的Servlet对象，如果没有则创建它，然后根据客户的请求创建HttpRequest、HttpResponse对象，从而调用Servlet对象的service方法。

3. Servlet

Servlet容器在启动时自动创建Servlet，这是由在web.xml文件中为Servlet设置的属性决定的。从中我们也能看到同一个类型的Servlet对象在Servlet容器中以单例的形式存在。

# 说说Servlet工作原理

Servlet的工作原理 :

1.首先简单解释一下Servlet接收和响应客户请求的过程，首先客户发送一个请求，Servlet是调用service()方法对请求进行响应的，通过源代码可见，service()方法中对请求的方式进行了匹配，选择调用doGet,doPost等这些方法，然后再进入对应的方法中调用逻辑层的方法，实现对客户的响应。在Servlet接口和GenericServlet中是没有doGet（）、doPost（）等等这些方法的，HttpServlet中定义了这些方法，但是都是返回error信息，所以，我们每次定义一个Servlet的时候，都必须实现doGet或doPost等这些方法。

2.每一个自定义的Servlet都必须实现Servlet的接口，Servlet接口中定义了五个方法，其中比较重要的三个方法涉及到Servlet的生命周期，分别是上文提到的init(),service(),destroy()方法。GenericServlet是一个通用的，不特定于任何协议的Servlet,它实现了Servlet接口。而HttpServlet继承于GenericServlet，因此HttpServlet也实现了Servlet接口。所以我们定义Servlet的时候只需要继承HttpServlet即可。

3.Servlet接口和GenericServlet是不特定于任何协议的，而HttpServlet是特定于HTTP协议的类，所以HttpServlet中实现了service()方法，并将请求ServletRequest、ServletResponse

强转为HttpRequest 和 HttpResponse。

Tomcat 与 Servlet 是如何工作的：

步骤：

1. Web Client 向Servlet容器（Tomcat）发出Http请求

2. Servlet容器接收Web Client的请求

3. Servlet容器创建一个HttpRequest对象，将Web Client请求的信息封装到这个对象中。

4. Servlet容器创建一个HttpResponse对象

5. Servlet容器调用HttpServlet对象的service方法，把HttpRequest对象与HttpResponse对象作为参数传给HttpServlet 对象。

6. HttpServlet调用HttpRequest对象的有关方法，获取Http请求信息。

7. HttpServlet调用HttpResponse对象的有关方法，生成响应数据。

8. Servlet容器把HttpServlet的响应结果传给Web Client。