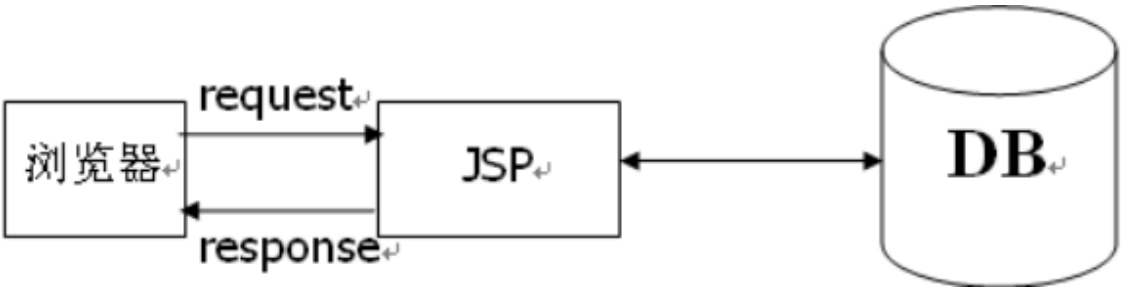
对比SpringMVC 中第一章的笔记

1. JavaWeb经历三个时期

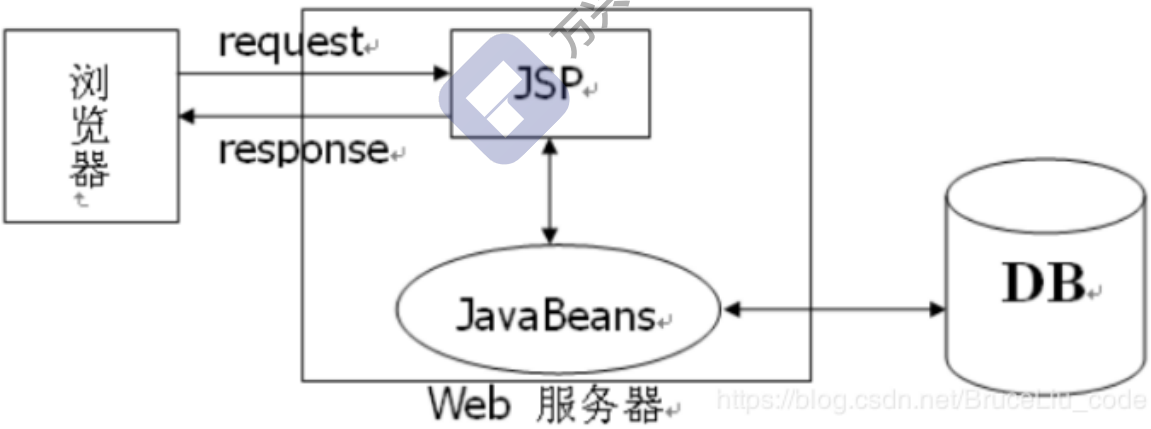
1.1.JSP Model1第一代

JSP Model1是JavaWeb早期的模型，它适合小型Web项目，开发成本低！Model1第一代时期，服务器端只有JSP页面，所有的操作都在JSP页面中，连访问数据库的API也在JSP页面中完成。也就是说，所有的东西都耦合在一起，对后期的维护和扩展极为不利。



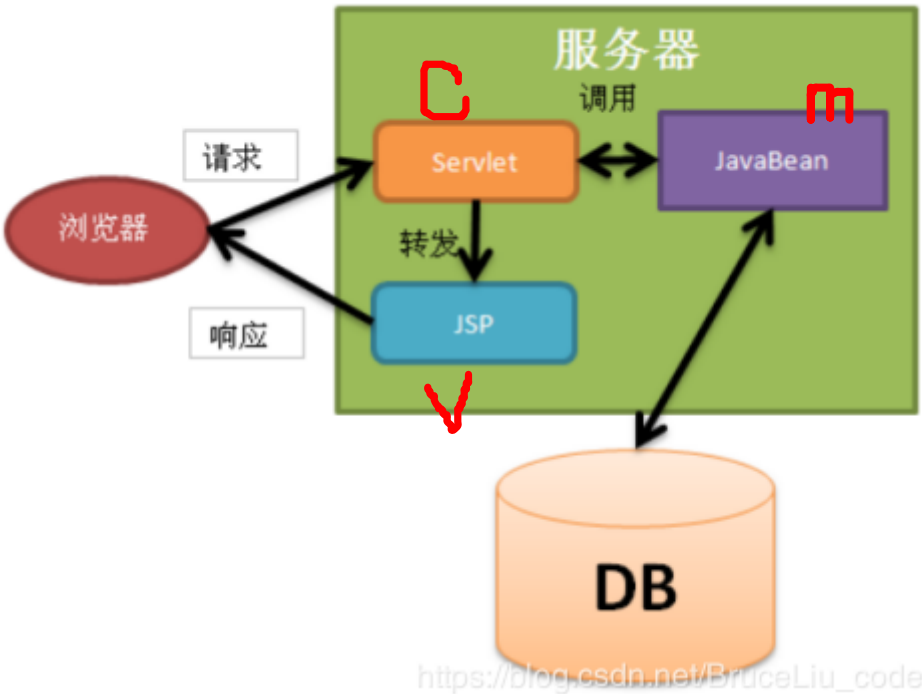
1.2. JSP Model1第二代

JSP Model1第二代有所改进，把业务逻辑的内容放到了JavaBean中，而JSP页面负责显示以及请求调度的工作。虽然第二代比第一代好了些，但还让JSP做了过多的工作，JSP中把视图工作和请求调度（控制器）的工作耦合在一起了。



1.3. JSP Model2

JSP：视图层，用来与用户打交道。负责接收用来的数据，以及显示数据结果；
Servlet：控制层，负责找到合适的模型对象来处理业务逻辑，转发到合适的视图；
JavaBean：模型层，完成具体的业务工作，例如：开启、转账等。



1.4. 小结

这就是javaweb经历的三个年代，JSP Model2适合多人合作开发大型的Web项目，**各司其职，互不干涉，有利于开发中的分工，有利于组件的重用**。但是，Web项目的开发难度加大，同时对开发人员的技术要求也提高了。

在我们的实际项目中有关MVC的结构在第三章的讲义中的Servlet实战.pdf的最后有项目中MVC的结构

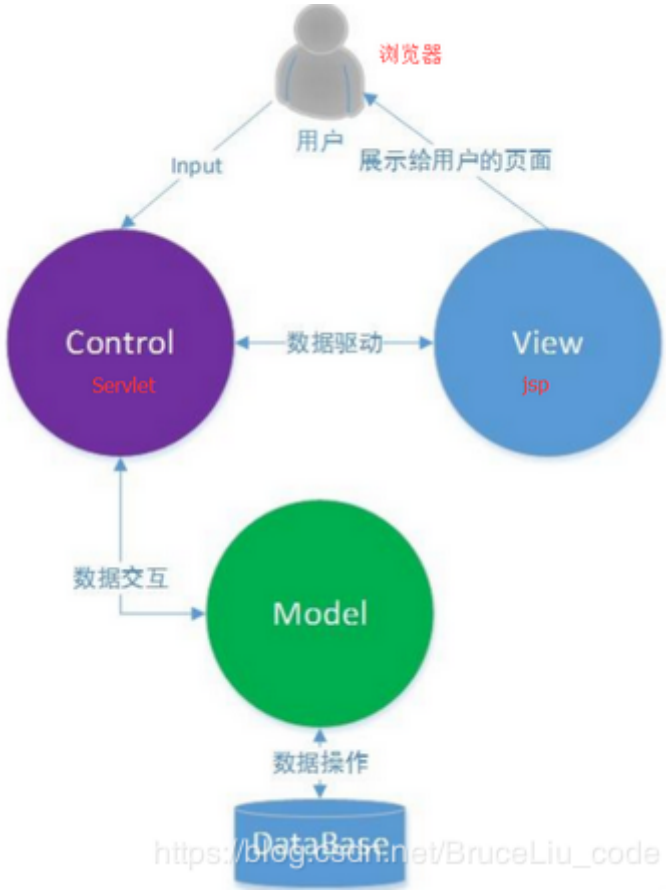
2. MVC概念

首先我们需要知道MVC模式并不是javaweb项目中独有的，MVC是一种软件工程中的一种软件架构模式，把软件系统分为三个基本部分：**模型（Model）、视图（View）和控制器（Controller）**，即为MVC。它是一种软件设计的典范，最早为Trygve Reenskaug提出，为施乐帕罗奥多研究中心（Xerox PARC）的Smalltalk语言发明的一种软件设计模式。

也就是前面几章中的Servlet

- **控制器(Controller)**：控制器即是控制请求的处理逻辑，对请求进行处理，负责请 求转发；
- **视图(View)**：视图即是用户看到并与之交互的界面，比如HTML（静态资源），JSP（动态资源）等等。
- **模型(Model)**：模型代表着一种企业规范，就是业务流程/状态的处理以及业务规则的规定。业务流程的处理过程对其他层来说是不透明的，模型接受视图数据的请求，并返回最终的处理结果。业务模型的设计可以说是MVC的核心。

  模型层在平时使用的时候会进行细分，例如：细分成–> 业务逻辑层 和 数据访问层；



就这个图来分析一波，画的确

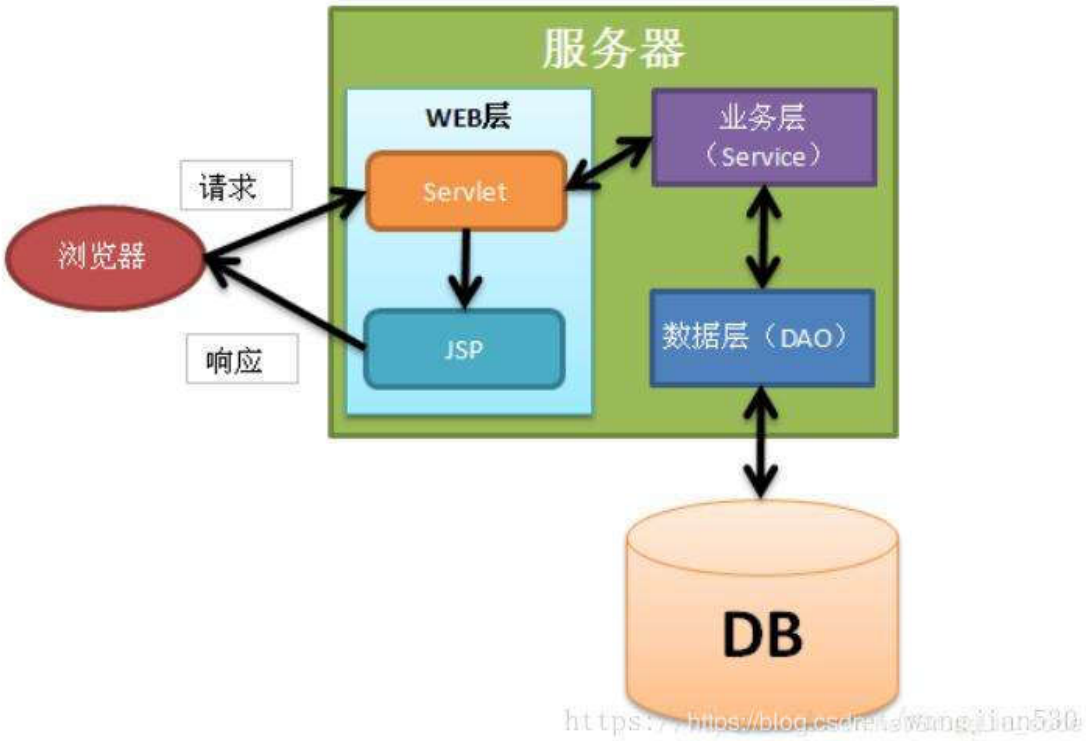
实挺不错的，首先，我们打开浏览器，输入网址，就是到服务器中请求页面(JSP也可能是别的)，然后显示到浏览器上，然后通过点击JSP页面上的内容，提交请求，到服务器中，也就到了Control(Servlet)这一块，Servlet通过分析请求，知道用户需要什么，需要数据，那么就通过Model，从数据库拿到数据，在将数据显示在JSP中，在将JSP发送回浏览器，显示在用户看，所以我们经常说，JSP就是View层，给用户看的，Servlet作为控制流程，而编写操作数据库代码，业务逻辑代码就属于Model。这就是MVC的应用。

3.JavaWeb经典三层架构 不同于MVC的分层

所谓三层是表述层（WEB层）、业务逻辑层（Business Logic），以及数据访问层（Data Access）。包括展示层和控制层

- **WEB层**：包含JSP和Servlet等与WEB相关的内容；
- **业务层**：业务层中不包含JavaWeb API，它只关心业务逻辑；

- **数据层**：封装了对数据库的访问细节；



实际开发中会抽象出业务逻辑层接口和数据访问层接口，方便后期使用Spring进行解耦，一般开发模式如下：