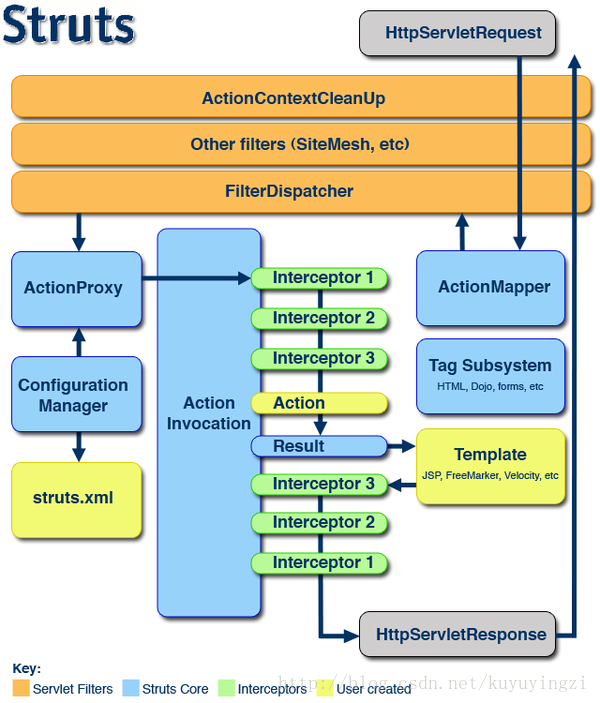
1. Struts：执行流程
   1. 
2. 执行流程文字描述：
   1. 在web.xml配置strutsPrepareAndExecuteFilter后，客户端的请求发送给此Filter。
   2. Filter询问ActionMapper：该请求是否是一个Struts2请求（即是否返回一个非空的ActionMapping对象）
   3. 若ActionMapper认为该请求是一个Struts2请求，则Filter把请求的处理交给ActionProxy
   4. ActionProxy通过Configuration Manager询问框架的配置文件（Struts.xml）,是否需要调用Action类及Action方法
   5. 之后ActionProxy创建一个ActionInvocation的实例，并且进行初始化
   6. ActionInvocation实例在调用Action的过程前后，涉及到相关拦截器的调用
   7. Action执行完毕，ActionInvocation负责根据Struts.xml中的配置找到对应的result。调用结果的excute方法，渲染结果。
   8. 执行各个拦截器inbvocation.invoke()之后的代码，然后把结果返回给客户端。
3. struts2是通过反射机制来执行action的方法，struts2规范目前action方法是不需要参数的，有参数的方法和没参数的方法是两个不同的方法，所以会出现找不到的情况。
4. Struts2基于注解配置时需要遵循一下规范
   1. Action所在的包名必须以.action结尾
   2. Action要继承ActionSupport类
5. Struts2加载常量的顺序是：Struts.xml，Struts.properties和Web.xml，如果这3个文件中对某个常量有重复配置时，后一个文件中配置的常量值会覆盖前面文件中的同名常量值。
6. 在Struts.xml中如果指定了该包为一个抽象包，则包内不能定义Action。
7. 在Struts.xml中配置Action时，如果没有指定class属性（即没有 提供用户的action类），系统自动使用ActionSupport类作为业务控制器Action。
8. Ognl标签不支持El表达式和<%=request.ContextPath%>写法，如需获得ContextPath则可以这么写%{#request.getContextPath()}。
9. Struts2访问Web资源有两种方式，第一种和Servlet API解耦的方法：使用ActionContext或者实现XXXAWare接口。第二种是和Servet API耦合的方式：继承ServletXXXAware接口或者使用ServletActionContext直接获取ServletAPI。
10. Session对应的Map实际上是sessionMap类型的，强转后调用其invalidate方法可使其失效。
11. Result的type属性常用的有：dispatcher（默认的：转发），redirect（重定向），redirectAction（重定向到一个Action），chain（转发到一个Action）
12. 转发和重定向（redirect）区别：转发是一次请求，一次响应。地址栏的url不会改变。在服务器端执行，所以必须在同一台服务器上。转发速度快。而重定向是两次请求。地址栏的url会随之改变为定向后的那个url。在客户端执行，所以可以在不同的服务器上。重定向速度慢。重定向之后request中的对象是不可用的，而转发可以。
13. 可以通过重定向（redirect）方式替代redirectAction，然而不能通过转发的方式替代chain。
14. 动态方法调用默认是false的。
15. Struts2从页面上获取的属性值并不是从请求域中获取的，而是从值栈中获取的。
16. ValueStack（值栈）：
    1. 是每个请求对象的轻量级的数据存储中心，在这里统一管理着数据，每一个请求创建一个新的值栈，所以值栈能够线程安全的为每个请求提供公共的数据存取服务。
    2. 可以从ActionContext中获取值栈对象
    3. 值栈分为两个逻辑部分:**ContextnlContext**类型，也是对ActionConext的一个引用，存放着各种Map，如：requestMap，sessionMap等。对象栈（ObjectStack）类型，是使用ArrayList定义的栈，存放这Action和相关的对象。
    4. 如果要访问ContextMap中的数据，则需要给OGNL表达式加上一个前缀字符#，否则搜索将在ObjectStack里进行。
17. Struts2自动把action对象放入到值栈当中，在调用action类的方法之前创建了StrutsActonproxy对象，对代理对象初始化时把action放入了值栈中。
18. Prepare拦截器执行流程：
    1. 若Action实现了Prepareble接口，则Struts将尝试执行prepare[方法名]方法
    2. 若prepare[方法名]方法不存在，将尝试执行prepareDo[方法名]方法。若都不存在，就都不执行
    3. 若prepareInterceptor的alawaysInvokePrepare属性为false（默认为true），则Struts2将不会调用实现prepareble接口的Action的prepare()方法
19. Struts2自定义类型转换有两种方式：基于字段的配置和基于类型的配置。
    1. 基于字段的配置：
       1. 在字段所在的Model（可能还一个Action或者是JavaBean）的包下，新建一个ModelClassName-conversion.properties文件
       2. 在该文件中输入键值对：fieldName=类型转换器的全类名
       3. 该方式是在第一次使用转换器式创建实例，且是单实例的。
    2. 基于类型的配置：
       1. 在src下新建xwork-conversion.properties文件
       2. 在该文件中输入键值对：待转换的类型=类型转换器的全类名
       3. 该方式在Struts2应用被加载时创建实例
20. 国际化资源文件加载的顺序：离当前Action较近的优先加载。
21. I18n拦截器负责将当前action请求的locale设置为session中指定的locale值。
22. 访问国际化资源文件的value值：
    1. 在页面上：
       1. Label标签使用%{getText（‘key值’）}，如%{getText(‘username’)}
       2. 使用key属性直接访问
    2. 在Action中
       1. Action中要实现TextProvider接口，该接口定义了访问国际化资源文件的getText（）方法，或者继承ActionSupport类（实现了TextProvider接口）
23. 使用中文资源文件时是需要把中文转化为Ascii码
24. 在输出key对应的value时，首先搜索包资源文件，如果没有匹配的key，则搜索全局文件。
25. 利用Struts进行文件的上传和下载：
    1. uploadFileFileName和uploadFileContentType两个属性的命名是以File类型的属性uploadFile作为基名
    2. FileModel一般有一下属性：File类型，文件的名称，文件的类型，文件的大小，文件在服务端的路径（一般在Struts.xml中设置）。
    3. 设置struts.multipart.maxSize可设置上传文件最大大小
    4. 多文件上传时可使用数据或者List集合

文件的下载：

1. 需要指定下载的类型：contentType
2. contentLength:用于指定流的长度（浏览器显示一个进度条），以字节为单位
3. contentDisposition：用于指定响应Content-Disposition头，指定下载文件的名称，有两个可选项：inline；filename=“下载文件的名称”，或者inline；filename=“下载文件的名称”。
4. bufferSize：用于设置下载文件是缓冲去的大小，默认1024字节
5. contentCharSet：设置字符集
6. inputName：用于指定action链中InputStream属性的名称，默认为inputStream。
7. 折腾了2个小时的错误：页面错误：multipart/form-data 写成了multipart/from-data
8. 进行数据的增删改时，执行完相关的操作后要提交事务，否则数据不会保存，在ssh整合之后因为声明式事务，所以无需手动提交。