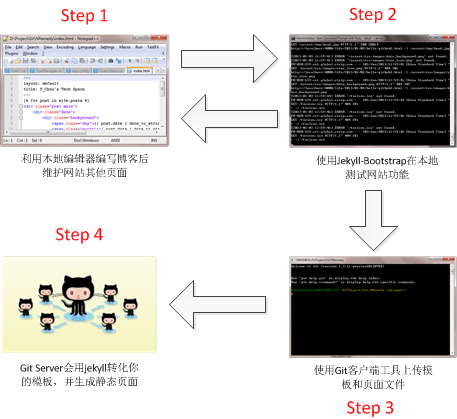
两种标记：

* {{ }}：此标记表征的是将其中的变量转化成文本
* {% %}：此标记用于包含控制流关键字，比如：{% if %}、{% for x in xx %}

维护流程

因此，对GitHub网站的维护工作，大致可以用下面的图表示：



1. 利用本地编辑器编写博客后维护网站其他页面
2. 使用Jekyll-Bootstrap在本地测试网站功能
3. 使用Git客户端工具上传模板和页面文件
4. Git Server会用jekyll转化你的模板，并生成静态页面

GitHub提供两种类型的主页

* 个人或组织主页
* 项目主页

在启动server后，之前我们的项目目录下会多出一个\_site目录。jekyll默认将转化的静态页面保存在\_site目录下，并以某种方式组织。

jekyll并不是博客软件，跟workpress之类的完全两码事，它仅仅是个一次性的模板解析引擎，它不能像动态服务端脚本那样处理请求。

在jekyll解析你的网站结构前，需要确保网站目录像下面那样：

|—— \_config.yml

|—— \_includes

|—— \_layouts

| |—— default.html

| |—— post.html

|—— \_posts

| |—— 20011-10-25-open-source-is-good.html

| |—— 20011-04-26-hello-world.html

|—— \_site

|—— index.html

|—— assets

|—— css

|—— style.css

|—— javascripts

* \_config.yml：保存配置，该配置将影响jekyll构造网站的各种行为。关于配置的详细文档在这里
* \_includes：该目录下的文件可以用来作为公共的内容被其他文章引用，就跟C语言include头文件的机制完全一样，jekyll在解析时会对{% include %}标记扩展成对应的在\_includes文件夹中的文件
* \_layouts：该目录下的文件作为主要的模板文件
* \_posts：文章或网页应当放在这个目录中，但需要注意的是，文章的文件名必须是YYYY-MM-DD-title
* \_site：上面提到过，这是jekyll默认的转化结果存放的目录
* assets：这个目录没有强制的要求，主要目的是存放你的资源文件，图片、样式表、脚本等。

对于基于静态页面的网站，你显然不希望每篇文章都要写html、head等与文章本身无关的重复的东西，那么容易想到的是将这些东西作为模板提取出来，以便复用，\_layouts文件夹中的文件可以作为这样的模板。现在我们在\_layouts文件夹中创建一个模板文件，default.html：

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<title>My blog</title>

</head>

<body>

{{content }}

</body>

<html>

default.html包含了每个html都需要的一些标记，以及一个个liquid标记。{{ … }}是liquid中用来表示“内容”的标记，其中的对象在解析时会被替换成文件到页面中

content：表示在这里的地方用**子页面（就是下面的index.html页面）**的内容替换。

现在我们来实现一个主页，在根目录下，创建一个index.html

---

layout: default

---

<h1>Hello jekyll</h1>

<p>This is the index page</p>

除了普通的html标记外，开头这一段称为[YAML格式](https://github.com/mojombo/jekyll/wiki/YAML-Front-Matter)，以一行---开头，一行---结尾，在虚线的中间以key-value的形式对一些全局变量进行赋值。

* site：全局站点对象。比如site.posts返回当前站点所有在\_post目录下的文章，上面的例子结合for循环来罗列所有的文章
* page：文章对象。比如page.url将返回page对象的url，上面的例子用该对象和属性返回了文章的链接

另外要补充的是site.baseurl，该值就是我们在\_config.yml中配置的baseurl啦！