Pug原名不叫Pug，原来是大名鼎鼎的jade，后来由于商标的原因，改为Pug，哈巴狗。以下是官方解释：

it has been revealed to us that "Jade" is a registered trademark, and as a result a rename is needed. After some discussion among the maintainers, "Pug" has been chosen as the new name for this project.

其实只是换个名字，语法都与jade一样。

1. Pug安装编译的二三事

开始安装之后

npm **install** -g pug

运行一下代码

pug **index**.pug

结果显示：

bash: **pu**g: **command** not found

找了网上很多内容和教程，大部分都是关于jade的，去Github看了一下官方的讨论区，才知道

You need to install pug-cli. The CLI was separated from **the** core library **as** part **of** this release.

所以,成功解决问题

npm install -g pug-**cli**

详情请看：[pug: command not found](https://github.com/pugjs/pug/issues/2295" \t "_blank)

二. 代码编辑器优化

1. sublime，可以在package control->install package中安装pug
2. webStrom，如果出现Invalid indentation,you can use tabs or spaces but not both错误，可以参考这篇文章[Jade报错：Invalid indentation,you can use tabs or spaces but not both问题  
   ]([http://blog.csdn.net/greenqin...](http://blog.csdn.net/greenqingqingws/article/details/9969579)) PS:学生可以凭借edu邮箱免费使用正版

三. 基础语法知识

一段简单的代码

doctype **html**

**html**

head

title hello pug

**body**

**h1** pug pug

使用命令：

pug -P -w **index**.pug

编译后的结果为：

**<!DOCTYPE html>**

<html>

<head>

<title>hello pug </title>

</head>

<body>

<h1>pug pug</h1>

</body>

</html>

1.类名和ID名

**a**.button

**a**(class="button")

编译后：

<a **class**="button"></a>

<a **class**="button"></a>

ID类似

2.属性

属性可以使用()包裹起来,属性值之间用逗号隔开  
比如

**a**(href="google.com",title="google")

3.文本内容

**a**(href="google.com",title="google") Hello World

多行文本内容

**p**.

asdfasdfa

asdfasd

adsfasd

asdfasdfa

或者

p

| dfas

| dfas

| dfas

文本含有标签

p

| dfas <strong>hey</strong>

| dfas

| dfas

或者

p

| dfas <strong>hey</strong>

strong hey man

| dfas

| dfas

4.注释

1. 单行注释

// just some paragraphs

<!-- just some paragraphs-->

1. 非缓冲注释

//- will not output within markup

不会被编译到HTML

1. 块注释

第一种

body

//

As much text **as** you want

can go here.

第二种

<body>

<!--

As much text as you want

can go here.

-->

</body>

1. IE注释

<!--[if IE 8]><html class='ie8'><[endif]-->

<!--[if IE 9]><html class='ie9'><[endif]-->

<!--[if IE ]><html class='ie8'><[endif]-->

5.变量

-var Pug="hello world"

title #{Pug}

转义

-var htmlData='<strong>sdf</strong>'

p#{htmlData}

p!=htmlData

非转义

-**var** htmlData='<strong>sdf</strong>'

**p** !{htmlData}

p=htmlData

编译前的代码

**p** \#{htmlData}

**p** \!{htmlData}

没有的变量赋值

p=data;

是空值而不是undefined

6.流程代码

-var array=[1,2,3,4]

-**for** (var k **in** imooc)

p=array[k]

-each k **in** array

p:#{k}

-**while**

-var array=[1,2]

**if** array.length>2

p true

**else**

p false

unless 为false，才执行，用法与if类似

-**var** array=[1,2]

unless !istrue

**p** hello

switch的功能

-**var** name='java'

case name

when 'java': **p** Hi,java

case name

when 'pug': **p** Hi,pug

default

**p** Hi

7.mixins

1.重复的代码块

mixin sayHi

**p** hello everyone

+sayHi

编译后

<p>hello everyone</p>

2.传入参数

mixin pet(name)

**li**.pet= name

**ul**

+pet('cat')

3.blocks

mixin article(title)

.article

h1= title

**if** block //是否有包含内容

block

**else**

**p** No content provided

+article('Hello world')

+article('Hello world')

**p** This is my

编译后：

<!--如果节点里面没有内容，就加上-->

<div class="article">

<h1>Hello world</h1>

<p>No content provided</p>

</div>

<div class="article">

<h1>Hello world</h1>

<p>This is my</p>

<p>Amazing article</p>

</div>

4.Attributes

mixin link(href, name)

//- attributes == {class: "btn"}

a(**class**!=attributes.class, href=href)= name

+link('/foo', 'foo')(**class**="btn")

attributes已经转义，所以应该使用！=避免二次转义  
编译后:

<a href="/foo" class="btn">foo</a>

5.不确定参数

mixin list(id, ...items)

ul(id=id)

each item **in** items

li= item

+list('**my**-list', 1, 2, 3, 4)

参数中要加入...，编译后：

<ul id="my-list">

<li>1</li>

<li>2</li>

<li>3</li>

<li>4</li>

</ul>