Spring is an Application Development Framework which provides you numerous out of the box features called modules. Few of the core features are listed below:  
  
Spring IoC  
Spring ORM  
Spring AOP  
Spring MVC  
Spring JDBC  
Spring OXM  
Spring JMS  
and many more...  
  
MVC is a J2EE design pattern which follows a layered architecture namely (Model, View and Controller).  Spring MVC is spring's implementation of the MVC design pattern.

MVC is a Spring module. You use it for designing web applications. MVC in Spring implements the front controller design pattern. In your web.xml you'll define a single servlet (DispatcherServlet) and all your requests will pass through it and will be attended by Controllers you will define (@Controller).

=====================================================================================

Spring framework can be visualized as an **alternative to, replacement for, or even addition to the Enterprise JavaBeans (EJB)** model as mentioned [here](http://en.wikipedia.org/wiki/Spring_Framework). That means a framework to develop distributed, scalable, secured, transactional application.

Spring MVC can be thought of as **an replacement to Struts, Stripes, Tapestry** etc, i.e. Presentation layer framework.

Spring framework is the super set of Spring MVC.

===================================================================================

Spring MVC is one component within the whole Spring Framework, to support development of web applications. You can use any web containers, so it works fine with Tomcat. Just think that Spring is a third party library. You just need to put the necessary jar files in the lib directory, and define a few xml files.

So basically when you just say Spring it is a just a framework. And by framework I mean lot of functionalities/jars. Like you may have core which has core functionality or aspectj etc. Spring MVC i.e model view controller is one such functionality offered by Spring framework. You can deffer processing of request to various controllers based on the pattern of URL requested.

**Spring MVC** is Spring Framework's own web framework that has been designed around **Dispatcher Servlet**, which intercepts the incoming requests and passes them on to the handlers, from where the controller takes care of the rest.

**Spring REST architecture**is also based on Spring MVC, slightly making the difference on the View part. Traditional Spring MVC relies on the View technology to render the model data, the Spring REST architecture also does the same, except that the model object is set directly into the HTTP response, which the **@ResponseBody** converts into JSON/XML automatically. The output of a RESTful web service has to be a JSON or an XML, a standard format that could be easily handled across different consumer application platforms.

In higher Spring versions, the **@RestController** can directly be used instead of**@Controller**and **@ResponseBody**.

**MVC** is a complete web application package, with a where you can write a view in JSP, controller in spring (@controller) and the model could be your pojo objects of db. In controller you can inject spring services like AOP, IOC etc. Where as spring rest gives you a rest template which can be used for any rest api implementation. In this case typically you don't have any dedicated view for your controller (@restcontroller) and return a json object. You can use this api in any application n any view.