Q1:

Yes, this limitation still exists in the autowired framework. The closest you can get to a new product is by using createspecial(). But even when you make something with createspecial, it still follows the interface so it would be a variation of an already existing product

Q2:

* 1. @factory instantiates each of the classes given
  2. @provider shows what factory is responsible for instantiating the class
  3. @autowired says that that the following is a field that needs to be injected
  4. @autowired(selector=”<some string switch>”) says that the following field needs injected and that it would like a specific class

Q3:

It is used to show which classes map to selector strings

Q4:

AbstractFactory.createSpecial() lets you instantiate specific subclasses that need a little more that the normal super class. For instance, the ny pizza sort need to be able to make veggie pizzas. This would normally not be possible since the veggie pizza also needs an array of veggies in addition to the dough, cheese, and sauce that normal pizzas need. createSpecial is used in the autowiring process to instantiate that veggie array

Q5:

WiredConfiguration is used to store exclusions to the autowire process. It implements a singleton pattern

Q6:

The wiredFramework is used to do the actual injections. It implements a singleton pattern