Keep in Mind

* JavaScript is an object-based language based on prototypes, rather than being class-based.

Class-based Vs. Prototype-based languages

* A class defines all of the properties (considering methods and fields in Java, or members in C++, to be properties) that characterize a certain set of objects. A class is an abstract thing. For example, the Employee class could represent the set of all employees.
* An instance, on the other hand, is the instantiation of a class; that is, one of its members. For example, Victoria could be an instance of the Employee class, representing a particular individual as an employee. An instance has exactly the same properties of its parent class (no more, no less).
* A prototype-based language has the notion of a prototypical object, an object used as a template from which to get the initial properties for a new object. Any object can specify its own properties, either when you create it or at run time. In addition, any object can be associated as the prototype for another object, allowing the second object to share the first object's properties.

| Comparison of class-based (Java) and prototype-based (JavaScript) object systems | |
| --- | --- |
| **Class-based (Java)** | **Prototype-based (JavaScript)** |
| Class and instance are distinct entities. | All objects can inherit from another object. |
| Define a class with a class definition; instantiate a class with constructor methods. | Define and create a set of objects with constructor functions. |
| Create a single object with the new operator. | Same. |
| Construct an object hierarchy by using class definitions to define subclasses of existing classes. | Construct an object hierarchy by assigning an object as the prototype associated with a constructor function. |
| Inherit properties by following the class chain. | Inherit properties by following the prototype chain. |
| Class definition specifies *all* properties of all instances of a class. Cannot add properties dynamically at run time. | Constructor function or prototype specifies an *initial set* of properties. Can add or remove properties dynamically to individual objects or to the entire set of objects. |