Contents

You may want to prevent application developers from creating new instances of certain entity types. You will need a "defense in depth" on both client and server to block the determined developer. In this topic we suggest that you **block entity construction** by making all constructors *internal*.

Fixed-set entity types

In every application there are "fixed-set" entity types. The number of states in the USA is fifty and that is not expected to change. If U.S. states are modelled as entities, you want to make it difficult to add a new state by accident.

You probably want to prevent modifications and deletions too, subjects covered elsewhere [LINK].

It's a good idea to add a **save interceptor rule** [LINK] that rejects any client's attempt to save a change to a U.S. State or to alter the set of U.S. states.

Unfortunately, a developer won't know that changes are forbidden until runtime when the application throws an exception. If you make the entity difficult to construct in the first place, the developer won't be tempted to make a mistake.

Write an internal default constructor

Consider writing an *internal* default constructor. Make all other constructors *internal* as well.

```
internal UsaState() { } // Do not create new states
```

```
Friend Sub New() ' Do not create new states
End Sub
```

In Silverlight application you **must** use *internal*, not *private* or *protected*. DevForce has to "materialize" UsaState objects retrieved by a query. Conceptually queries *reconstitute* existing entities; they don't *create* new entities. In practice, "materialization" may involve calling a class constructor. DevForce can call an entity's *internal* constructor; it can't call a *private* or *protected* constructor in Silverlight.

Leave a back-door

Make it difficult to create an entity but not impossible. You don't want to query for test entities and you can't query for design entities. You need a back-door to create them in memory and make it look like you queried for them - a technique covered elsewhere [LINK]. The *internal* constructor may be sufficient.

Alternatively, you can add a static *CreateForTest* method to the entity class that is clearly limited in intent. Surround it in compiler directives to ensure it doesn't show up in your release builds.

```
#if DEBUG
  ///<summary>Internal use only</summary>
  public static UsaState CreateForTest
   {
     return new UsaState();
   }
#endif
```

```
#If DEBUG Then
 '''<summary>Internal use only</summary>
 Public Shared ReadOnly Property CreateForTest() As UsaState
 Return New UsaState()
 End Property
#End If
```