

## Contents

- [Using "new"](#)
- [Introducing CreateEntity\(...\)](#)
- [The hidden EntityManager](#)
- [Generalizing entity creation with the non-generic overload](#)
- [Conclusion](#)

While we recommend using "new" to create the entity within a factory method, the [CreateEntity](#) factory is available and merits some consideration. This topic describes it, when you might want to use it, and why you probably don't.

## Using "new"

We generally recommend that you call a constructor within a factory method when creating a new entity ... as we typically do throughout this documentation. Here's a simple factory method that uses "new".

```
public Customer CreateCustomer()
{
    return new Customer(); // more to follow
}
```

```
Public Function CreateCustomer() As Customer
    Return New Customer() ' more to follow
End Function
```

## Introducing CreateEntity(...)

DevForce offers an alternative approach that uses the *CreateEntity* factory method of the [EntityManager](#). Here's the same example - or close to it - written with *CreateEntity*:

```
public Customer CreateCustomer(EntityManager manager)
{
    return manager.CreateEntity<Customer>(); // more to follow
}
```

```
Public Function CreateCustomer(ByVal manager As EntityManager) As Customer
    Return manager.CreateEntity(Of Customer)() ' more to follow
End Function
```

Although this approach requires the help of an EntityManager, it doesn't actually add the new Customer to the manager. You have to do that in a separate step. It differs substantively from "new" in these respects:

1. It uses the default constructor internally to instantiate the entity
2. It hides a reference to the EntityManager inside the created entity.

Many of us fail to see the advantage of these differences.

- It's often inappropriate or impossible to create a new entity with a default constructor as discussed in the topic on [writing a custom constructor](#).
- The embedded EntityManager is hidden, silent, and (mostly) inaccessible until after the entity is added to cache.

## The hidden *EntityManager*

The *CreateEntity* method embeds an *EntityManager* within the new entity. Its presence enables the following technique for adding the entity to the *EntityManager*:

```
cust.EntityAspect.AddToManager();
```

```
cust.EntityAspect.AddToManager()
```

You can't call [AddToManager](#) on an entity you created with "new". You have to write:

```
manager.AddEntity(cust);
```

```
manager.AddEntity(cust)
```

The embedded *EntityManager* is otherwise inaccessible as demonstrated in these tests:

```
var cust == manager.CreateEntity<Customer>(); // hides an EntityManager inside the entity
```

```
Assert.IsNull(cust.EntityAspect.EntityManager); // EntityManager is not visible
cust.EntityAspect.AddToManager();             // adds the entity to the hidden EntityManager
Assert.IsNotNull(
    cust.EntityAspect.EntityManager());        // now you see it.
```

```
Dim cust = manager.CreateEntity(Of Customer)() ' hides an EntityManager inside the entity
Assert.IsNull(cust.EntityAspect.EntityManager) ' EntityManager is not visible
cust.EntityAspect.AddToManager() ' adds the entity to the hidden EntityManager
Assert.IsNotNull(cust.EntityAspect.EntityManager()) ' now you see it.
```

## Generalizing entity creation with the non-generic overload

The non-generic overload of CreateEntity has potential in a few scenarios. It could be convenient if you were writing a general utility that created entities as one of its duties.

```
public object CreateUserSelectedEntity(EntityManager manager, Type entityType)
{
    var anEntity = manager.CreateEntity(entityType);
    // do something with it
    return anEntity;
}
```

```
Public Function CreateUserSelectedEntity(ByVal manager As _
    EntityManager, ByVal entityType As Type) As Object
    Dim anEntity = manager.CreateEntity(entityType)
    ' do something with it
    Return anEntity
End Function
```

It is modestly difficult to "new" a class when you don't know its type. The DevForce *CreateEntity* method does it without ceremony.

## Conclusion

Stick with "new" unless you are writing an entity creation utility.