

Contents

- [Instance Validation](#)
- [EntityServerSaveInterceptor](#)

By default, any changes submitted via an [EntityManager.SaveChanges](#) call are **validated on the server** as part of the save [life cycle](#).

Instance Validation

DevForce will automatically run instance verification on each entity in the save list.

If there are any validation errors an [EntityManagerSaveException](#) is returned to the client. The [FailureType](#) on the exception will be set to [PersistenceFailure.Validation](#), and the error message will be a concatenation of all validation error messages. The [EntitiesWithErrors](#) collection on the exception will also contain the first of any entities which failed validation.

The "EntitiesWithErrors" is something of a misnomer because only the first error encountered is currently returned. We did this so that we could add a switch to DevForce at a later date that would cause all entities to be validated before returning and would not need to change, and thus create a breaking change, to EntityManagerSaveException api as a result. The [EntityAspect.ValidationErrors](#) property on the 'failed' entity will also contain the validation errors returned from the server. If you are binding to these entities in your UI the validation error(s) are displayed as they would with local validation failures (either on the property or instance).

Be sure to wrap a *SaveChanges* or *SaveChangesAsync* call in a try/catch

EntityServerSaveInterceptor

Server validation is performed by the [EntityServerSaveInterceptor](#), which you can sub-type to customize save processing.

The default *VerifierEngine* used is a single instance of a thread safe "concurrent" [VerifierEngine](#). Verifiers are discovered and added to the server-side engine in the same way as they are in the client application, but it is not required that the same verifiers be defined on both client and server. In this way you can define verifiers which will only be executed on the client, or only executed on the server.

To modify server-side validation, sub-class the *EntityServerSaveInterceptor* and override the [ValidateSave](#) virtual method.