# Context Injection

he following example defines a context class to store referred books. The context class is injected to a binding class.

public class CatalogContext

{

public CatalogContext()

{

ReferenceBooks = new ReferenceBookList();

}

public ReferenceBookList ReferenceBooks { get; set; }

}

[Binding]

public class BookSteps

{

private readonly CatalogContext \_catalogContext;

public BookSteps(CatalogContext catalogContext)

{

\_catalogContext = catalogContext;

}

[Given(@"the following books")]

public void GivenTheFollowingBooks(Table table)

{

foreach (var book in table.CreateSet<Book>())

{

SaveBook(book);

\_catalogContext.ReferenceBooks.Add(book.Id, book);

}

}

}

In the first example we define a POCO for holding the data of a person and use it in a given and a then step that are placed in different binding classes.

public class PersonData // the POCO for sharing person data {

public string FirstName;

public string LastName;

}

[Binding]

public class MyStepDefs

{

private readonly PersonData personData;

public MyStepDefs(PersonData personData) // use it as ctor parameter

{

this.personData = personData;

}

[Given]

public void The\_person\_FIRSTNAME\_LASTNAME(string firstName, string lastName)

{

personData.FirstName = firstName; // write into the shared data

personData.LastName = lastName;

//... do other things you need

}

}

[Binding]

public class OtherStepDefs // another binding class needing the person

{

private readonly PersonData personData;

public OtherStepDefs(PersonData personData) // ctor parameter here too

{

this.personData = personData;

}

[Then]

public void The\_person\_data\_is\_properly\_displayed()

{

var displayedData = ... // get the displayed data from the app

// read from shared data, to perform assertions

Assert.AreEqual(personData.FirstName + " " + personData.LastName,

displayedData, "Person name was not displayed properly");

}

}