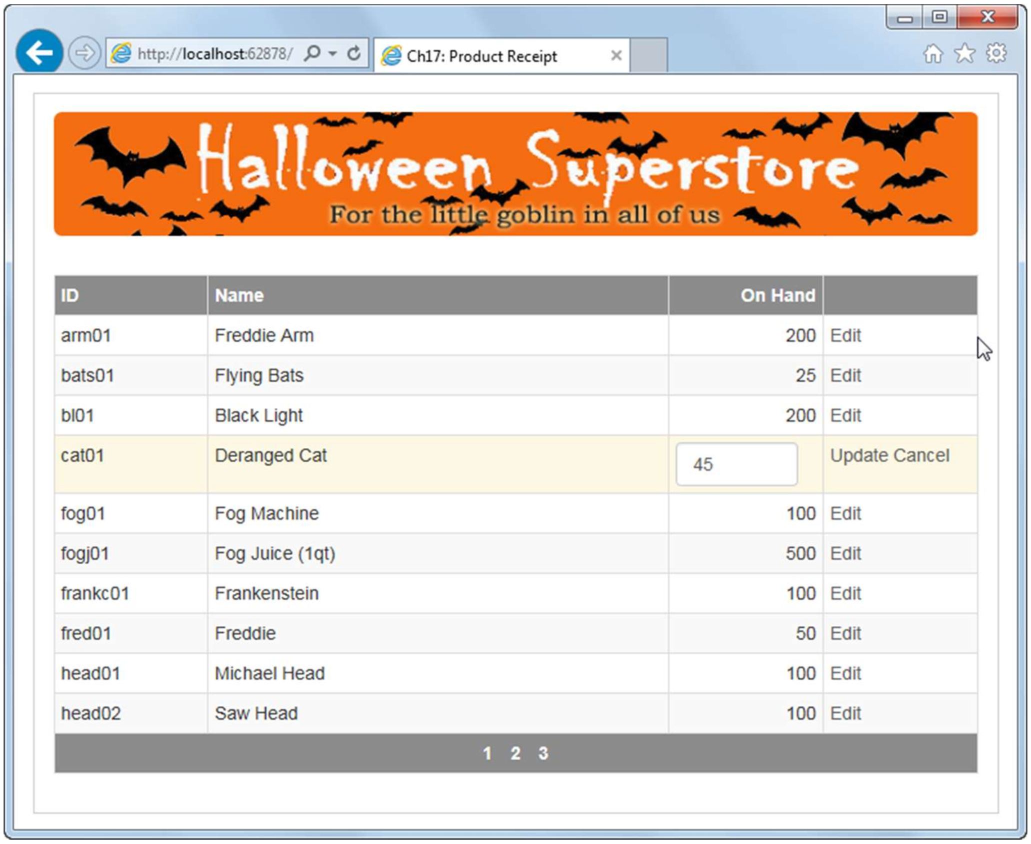
Modify the Product Receipt application

In this exercise, you'll modify a Product Receipt application so it uses an object data source instead of a SQL data source.



ID	Name	On Hand	
arm01	Freddie Arm	200	Edit
bats01	Flying Bats	25	Edit
bl01	Black Light	200	Edit
cat01	Deranged Cat	45	Update Cancel
fog01	Fog Machine	100	Edit
fogj01	Fog Juice (1qt)	500	Edit
frankc01	Frankenstein	100	Edit
fred01	Freddie	50	Edit
head01	Michael Head	100	Edit
head02	Saw Head	100	Edit

Review the starting code

1. Open the XEx17ProductReceipt application.
2. Review the aspx code for the Default page. Then, run the application to see how it works.

Build Business Object class and Data Access Class

3. Code the Product class in the Models folder. It defines three properties for the ProductID, Name, and OnHand columns.
4. Code the ProductDB class. Implement GetProducts method that returns a List<Product> object and the UpdateProduct method accepts a single Product object since it doesn't implement optimistic concurrency.

Add an ObjectDataSource control

5. Add an ObjectDataSource control to the Default page, and configure it so it retrieves data using the GetProducts method of the ProductDB class and updates a product using the UpdateProduct method. When you're done, make sure that the object data source is configured to use the Product business class to pass values to the update method.

6. Change the data source for the GridView control from the SQL data source to the object data source you just created. When a dialog box is displayed asking if you want to regenerate the GridView column fields and data keys, click No.

7. Delete the SqlDataSource control since it's no longer used by the page.

8. Run the application and test it to be sure it works correctly.

Use the designer to review the bound fields

9. Display the Fields dialog box for the GridView control. Then, select the ID field, and scroll through the properties for that field until you can see the DataField property.

10. Display the drop-down list for the DataField property, and notice that you can see the names of all the available fields. That's because the select method returns a strongly-typed List<Product> collection.