# Working with Service Fabric Services – Part II

## Introduction

Estimated time to complete this lab

60 minutes

After completing this lab, you will be able to:

* Understand the concepts of Service Fabric manifests
* Understand the concepts of Service Fabric stateful services, Stateless Gateway pattern

## Prerequisites

Before working on this lab, you must have the below prerequisites met. See [Prepare you development environment](https://azure.microsoft.com/en-us/documentation/articles/service-fabric-get-started/) for information on how to install a development environment on your machine.

* [Visual Studio 2015](https://www.visualstudio.com/en-us/products/vs-2015-product-editions.aspx) (with Update 3)
* [Service Fabric SDK](http://www.microsoft.com/web/handlers/webpi.ashx?command=getinstallerredirect&appid=MicrosoftAzure-ServiceFabric) (v2.1 or higher)
* Competed Part I of this lab or download code as a starting point.

## Overview of the lab

Goal of this lab is to make you familiar with an end to end development flow for Service Fabric applications. You will practice adding a stateful service to the stateless voting service fabric that was started in lab one.

## Scenario

<TODO>

There is a bug in the first part of the lab that didn’t affect the flow, we’ll address that first

## Reliable Dictionaries and Transactions

1. Open VotingService.cs and add using directives for **System.Net** and **System.Text**.
2. Just below the CreateServiceReplicaListeners method, insert the following code which processes each HTTP client request. Then, replace the TODO comments with actual code that does the operations as described.

private async Task ProcessRequestAsync(HttpListenerContext context, CancellationToken ct)

{

String output = null;

try

{

// Grab the vote item string from a "Vote=" query string parameter

HttpListenerRequest request = context.Request;

String voteItem = request.QueryString["Vote"];

if (voteItem != null)

{

// TODO: Here, write code to perform the following steps:

// Hint: See the RunAsync method to help you with these steps.

// 1. Get a reference to a reliable dictionary using the

// inherited StateManager. The dictionary should String keys

// and int values; Name the dictionary “Votes”

// 2. Create a new transaction using the inherited StateManager

// 3. Add the voteItem (with a count of 1) if it doesn’t already

// exist or increment its count if it does exist.

// The code below prepares the HTML response. It gets all the current

// vote items (and counts) and separates each with a break (<br>)

var q = from kvp in voteDictionary.CreateEnumerable()

//orderby kvp.Key // Intentionally commented out

select $"Item={kvp.Key}, Votes={kvp.Value}";

output = String.Join("<br>", q);

}

}

catch (Exception ex) { output = ex.ToString(); }

// Write response to client:

using (var response = context.Response)

{

if (output != null)

{

Byte[] outBytes = Encoding.UTF8.GetBytes(output);

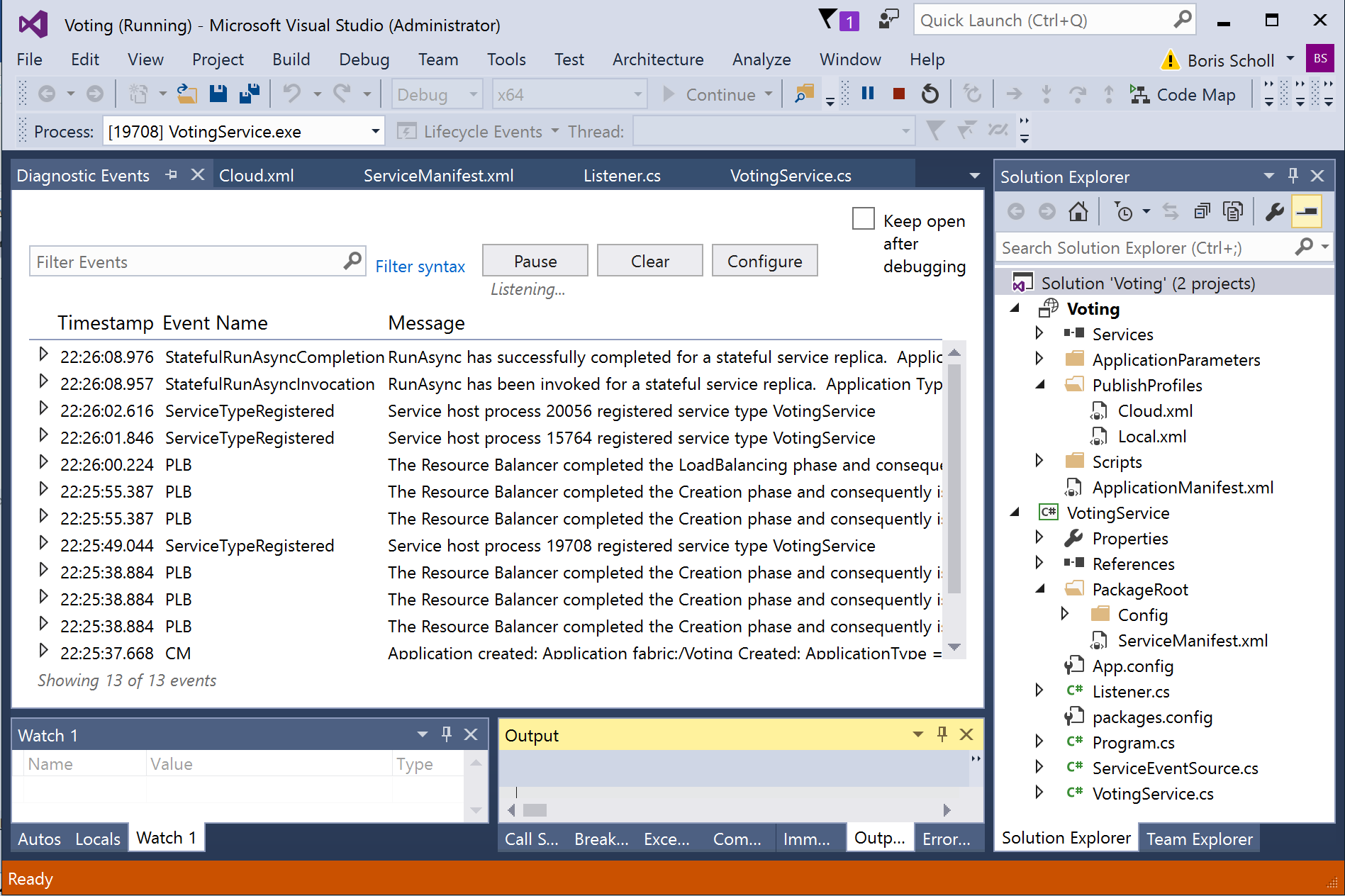
response.OutputStream.Write(outBytes, 0, outBytes.Length);

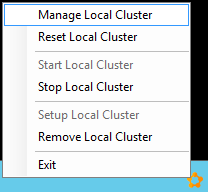
}

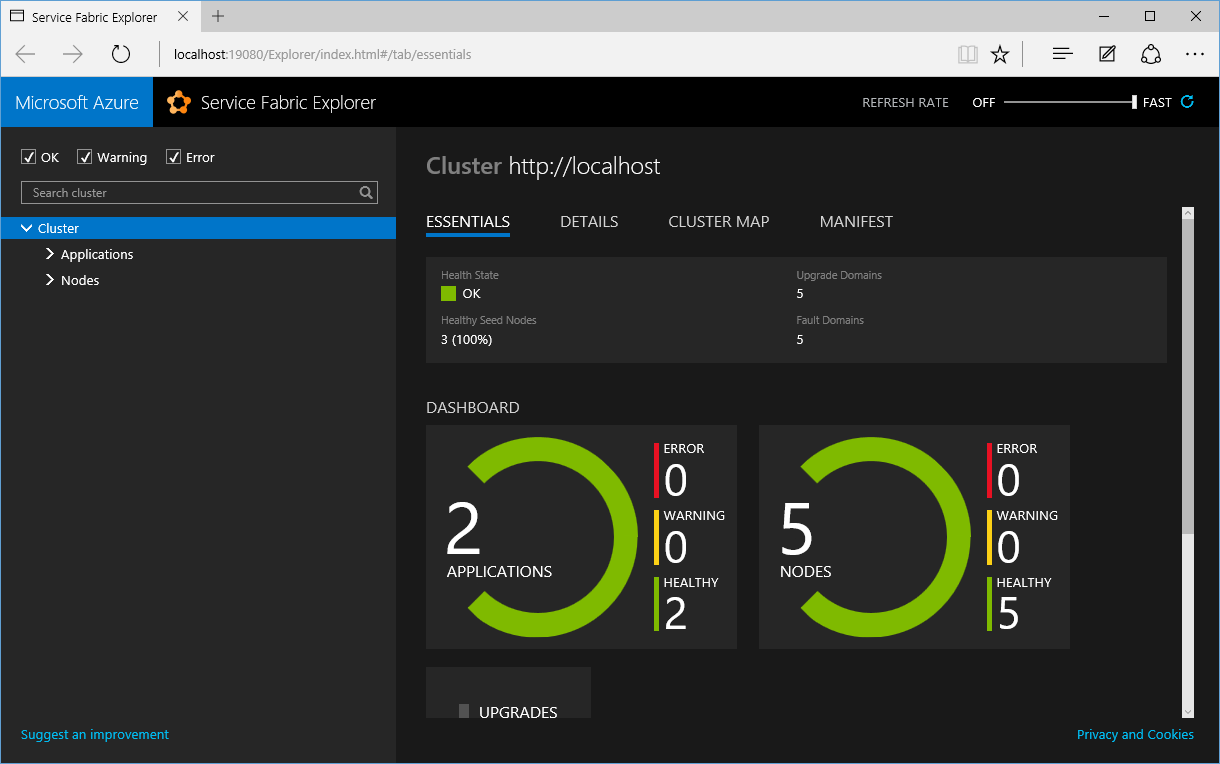
}

}

1. Comment out the RunAsync method in its entirety as it is not needed in this lab.
2. Hit F5 to deploy the application to the local development cluster
3. The application is running when you see events in Visual Studio’s Diagnostics Events. (If you don’t see the window, open it by using View🡪Other Windows🡪Diagnostics Event Viewer)

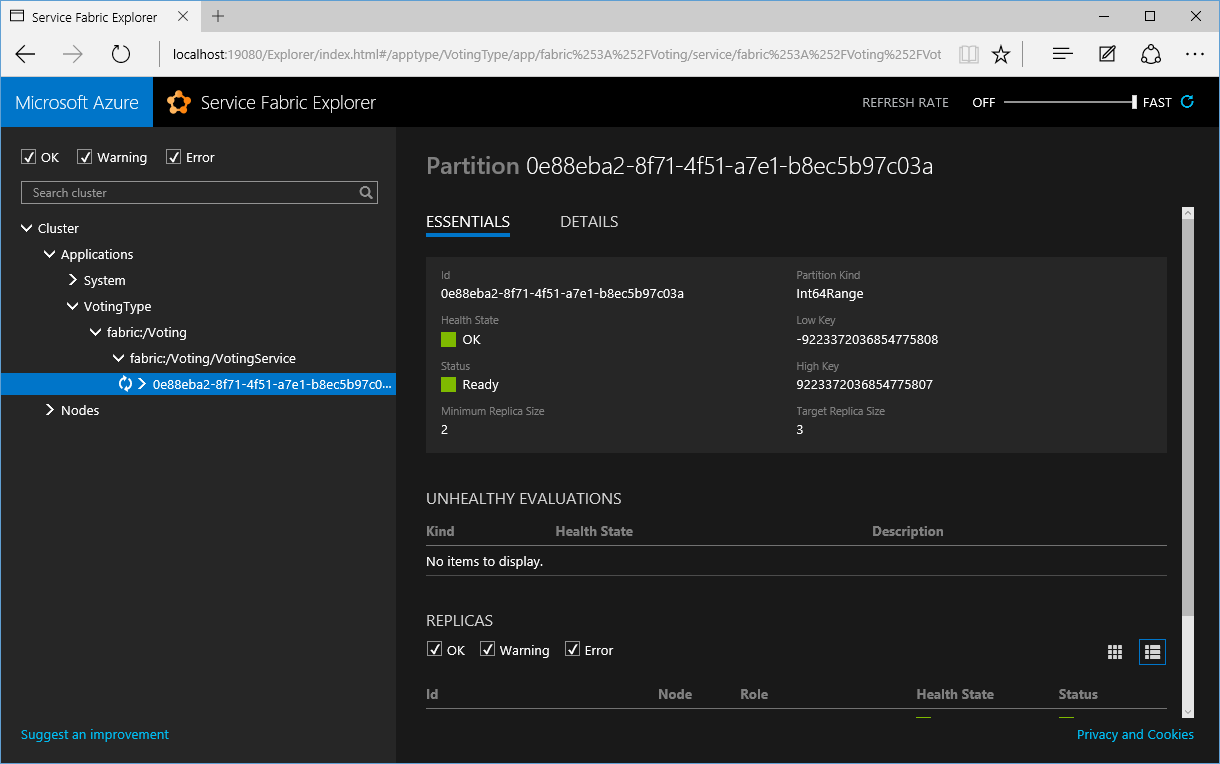


1. Right-click the Local Cluster Manager system tray app and choose Manage Local Cluster to launch Service Fabric Explorer.  
   
2. Service Fabric Explorer provides an overview of what’s happening in the cluster. It shows two application types, one for the Service Fabric system services and one for your VotingType.

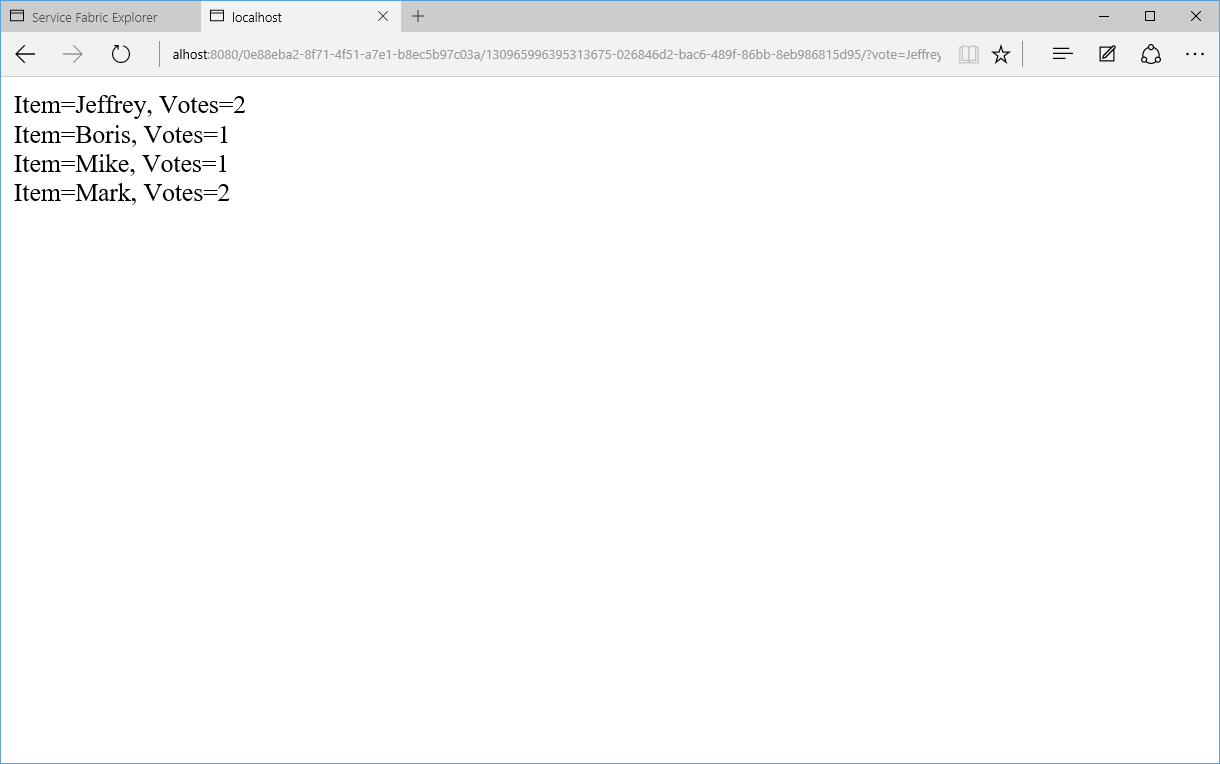


## Replica endpoints

1. Expand the Applications section on the left hand side and click on the VotingType entry.
2. Click on the fabric:/Voting entry. This uri is the name for the running instance of the application. We refer to this as a named application.
3. Expand the fabric:/Voting entry to see the named service running within it (fabric:/Voting/VotingService). You see only 1 entry here because you’re running only 1 instance of your service type within the named application.
4. Expand the fabric:/Voting/VotingService named service entry to see the partition Id running within it. Partition Ids are GUIDs. In the Figure below the partition Id is 0e88eba2-8f71-4f51-a7e1-b8ec5b97c03a



1. Expand the partition Id entry to see the nodes on which each replica resides.
2. Click on the entry for the node that holds the partition’s primary replica and within the Address section, copy the entire endpoint (starting with “Http://...”) to the clipboard. For your information, the endpoint consists of the following components:
   1. http: the scheme used to talk to the replica (as defined by our listener)
   2. Localhost: the IP address of the node
   3. 8080: the port the listener is listening to (as specified in the ServiceManifest.xml file)
   4. Guid: the partition Id (this helps diagnose problems; the URL itself indicates the partition you’re trying to access)
   5. 64-bit integer: the replica Id (this helps diagnose problems; the URL itself indicates the replica you’re trying to access)
   6. Guid: A random guid used to ensure the URL endpoint is unique
3. Paste the endpoint into a browser, remove the backslashes, and append a query string parameter “?vote=***SomeString***” (replace ***SomeString*** with a value of your choice). The address should look similar to: <http://localhost:8080/0e88eba2-8f71-4f51-a7e1-b8ec5b97c03a/130965996395313675-026846d2-bac6-489f-86bb-8eb986815d95/?vote=Jeffrey>. Sending this request is how you vote for something; in this case, “Jeffrey”
4. Vote for “Jeffrey” a few times (by refreshing your browser tab) and feel free to change the string you’re voting for as well. Here’s a figure showing a few votes for Jeffrey, Boris, Mike, and Mark:

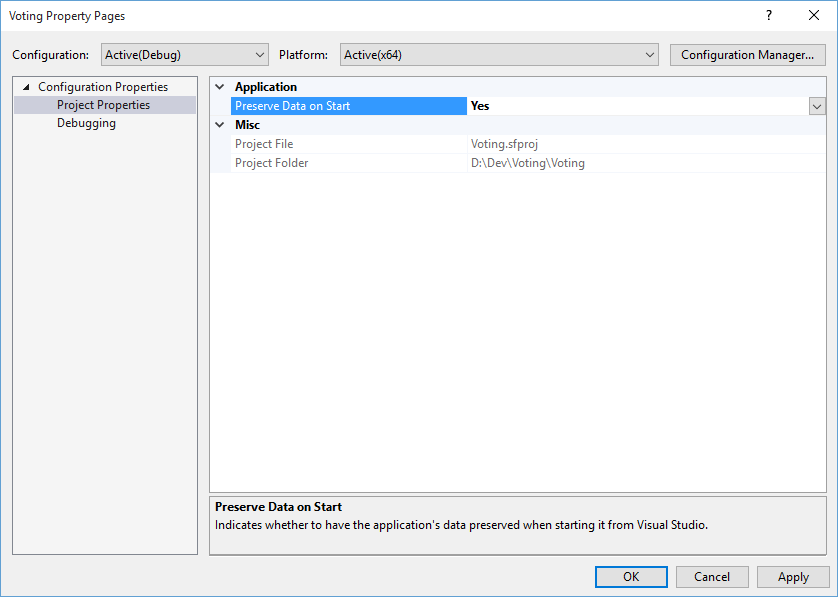


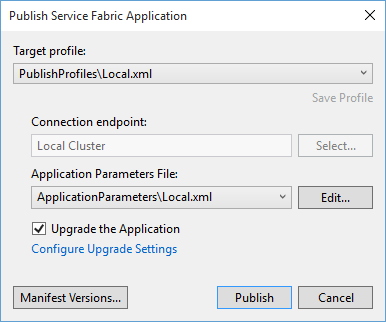
1. Go back to Visual Studio and stop the debugging session.

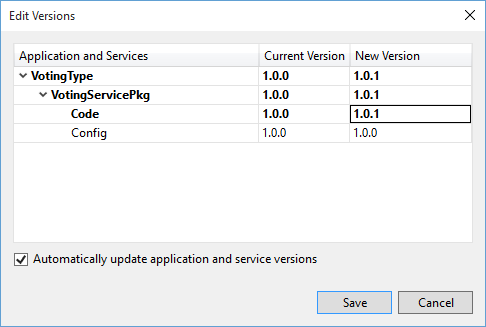
## Upgrade the named application

In this section you will learn how to upgrade the Voting application without any downtime and without any loss of data. (This is what Service Fabric is all about folks!!!)

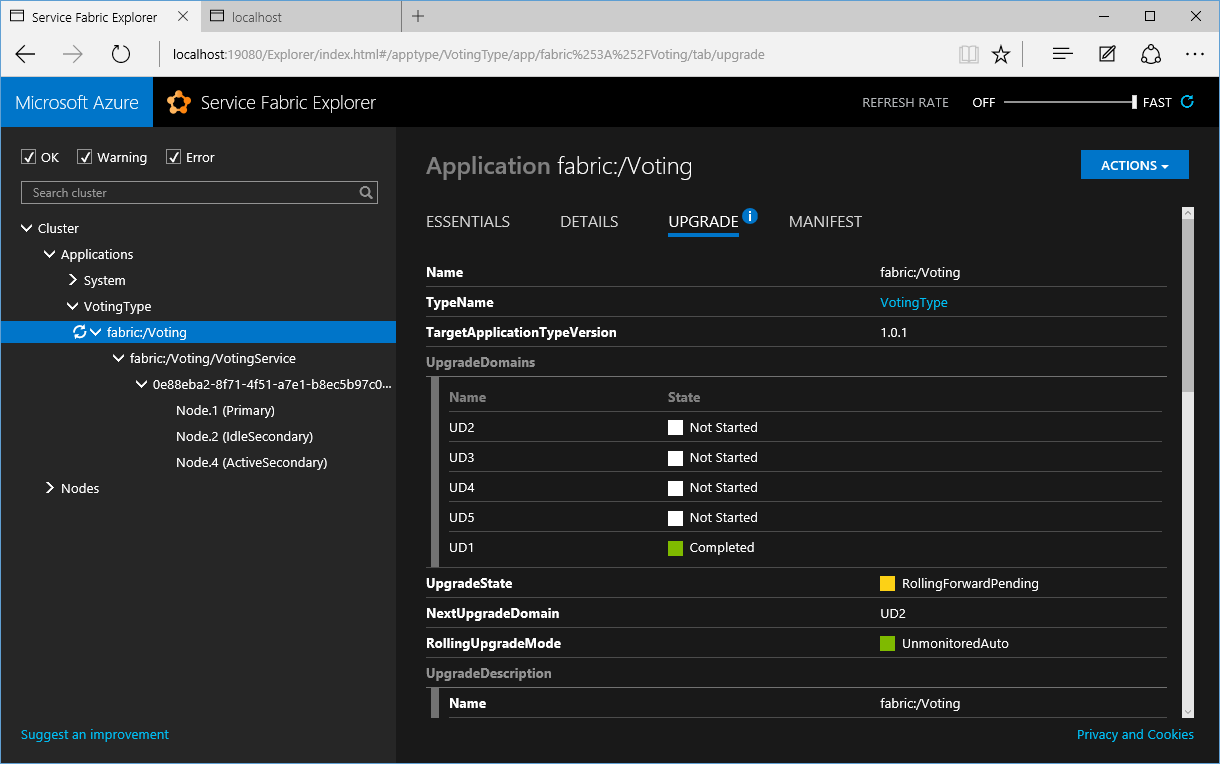
1. Inside the VotingService.cs file, locate the commented out statement //orderby kvp.Key in the ProcessRequestAsync method and uncomment it.
2. In Visual Studio Solution Explorer right click on the Voting project and select Properties. In the Voting Property Pages dialog change Preserve Data on Start to Yes as shown below. This setting allows you to upgrade your service in the local cluster when you hit F5 in Visual Studio as opposed to deleting it and starting it fresh. The difference is that an upgrade preserves any data while delete and start fresh destroys any data.



1. After uncommenting out the orderby code, you can test your new code by hitting F5 to start debugging or, you could run the code without the debugger by right-clicking on the Voting project and selecting Publish. This shows the following dialog:  
   
2. Before you publish, you **must** change the version number of your application type’s package, service type’s package, and code package by clicking on the Manifest Versions button.
3. Change the Code’s new version to 1.0.1 (this changes VotingType and VotingServicePkg version numbers automatically)



1. Hit Save followed by Publish
2. Go back to the Service Fabric Explorer, select the fabric:/Voting named application and select its Upgrade tab (on the right). Here, you’ll see the upgrade rolling through the cluster’s Upgrade Domains (UDs). Note: it usually takes a few minutes until al UDs have completed their upgrade.



1. After all UDs have completed their upgrade, you can see the application package’s new version in the Essentials tab
2. Click on the entry for the node that holds the primary replica and copy the http endpoint to the clipboard. Note: This endpoint will be different from the earlier one because replia IDs change during an upgrade.
3. Paste the new endpoint into a browser, remove the backslashes and append a query string parameter “?vote=***SomeString***” with a value of your choice. After you execute this, the response should show all the previous vote data since the data was preserved. In addition, you should now see that the vote strings in sorted order since you updated your code.

**Optional:** Modify your code again to sort the votes by vote count and upgrade the service within the cluster.

## Failover and PowerShell

1. Open Service Fabric Explorer and check which node holds your named service’s primary replica by expanding its partition entry.
2. Open PowerShell ISE and connect to the local Service Fabric cluster by entering the command below

Connect-ServiceFabricCluster

1. In PowerShell ISE, remove the primary replica (SF will automatically elect a secondary replica to become the new primary replica).

Remove-ServiceFabricReplica -ServiceName fabric:/Voting/VotingService  
 –ReplicaKindPrimary

1. Repeat steps 24 and 25 to enter the endpoint information for the new primary
2. The vote results should still show the same values.

## Optional

* Deploy and test the Voting application to the secure cluster you created at the beginning of the lab.
* Use more than one partition