BlackBerry

BlackBerry Developer Summit

Rapid Development Leveraging BEMS Services and Shared Services Framework - iOS

Table of Contents

1. Workbook Scope

This workbook corresponds to the session –

- Rapid Development Leveraging BEMS Services and the Shared Services Framework

2. Compatibility

These exercises can't be done with an application running in Enterprise Simulation mode.

3. Source code download & setup

Application sample code can be downloaded from –

https://developers.blackberry.com/us/en/resources/developer-summit/beginner-sessions-archive.html

The sample application depends on the BlackBerry Dynamics SDK. Ensure BlackBerry Dynamics SDK is installed. SDK version 4.x has been tested with this sample

When installed onto an iOS Device or Simulator the sample code app name is 'Services'

4. Shared Services Exercises

These exercises can't be done with an application running in Enterprise Simulation mode. If necessary, see the Application developer basics manuals for instructions on running applications in normal enterprise mode.

The Application-Based Services feature is for communication between two BlackBerry Dynamics applications, referred to as the service *consumer* and the service *provider*. You will therefore need two applications in order to complete these exercises.

The keyboard on the Services sample application can be dismissed by tapping on the screen or by pressing the return button from a TextField(not a TextView).

Exercise considerations – BlackBerry Work

One way to do these exercises is to use BlackBerry Work as one of the applications. BlackBerry Work is a provider of services, including Send Email and Transfer File, and also a consumer of services, including Transfer File.

Note that BlackBerry Work can only be installed onto a physical device from App Store. If you are running iOS Simulator, activate and run the AppKinetics sample app from the location below:

```
~/Library/Application Support
/BlackBerry/Good.platform/iOS/Examples/objective-c/
```

If you are using XCode10. There are two ways to make existing apps compatible with XCode 10.

1. Change path in the xcconfig file. Update the existing path for LDPLUSPLUS and LD from
   LDPLUSPLUS=~/Library/Application Support/BlackBerry/Good.platform/iOS/FIPS_module/$FIPS_PACKAGE/bin/g d_fipsld
   LD=~/Library/Application Support/BlackBerry/Good.platform/iOS/FIPS_module/$FIPS_PACKAGE/bin/g d_fipsld
   to
   LDPLUSPLUS=$(HOME)/Library/Application Support/BlackBerry/Good.platform/iOS/FIPS_module/$FIPS_PACKAGE/bin/g d_fipsld
   LD=$(HOME)/Library/Application Support/BlackBerry/Good.platform/iOS/FIPS_module/$FIPS_PACKAGE/bin/g d_fipsld

   Note the change from ~ to $(HOME).

2. Switch to Legacy Build System. You can switch to Legacy Build System in XCode by going to File>Project Settings>Build System

4.1. Exercise One: Service Provider Discovery

Select a service to consume.

A complete list of services can be found on the application developer portal.

https://apps.good.com/#/services

This exercise doesn't cover server-based services.

For example, you could select the Transfer File service.

1. Get the service identifier and version.

   The service listing that is mentioned in the previous step shows all the identifiers. Pages linked from the listing show the service versions, and the details of the service. Most services have only a single version.

   For the Transfer File service:

   - The service identifier is: `com.good.gdservice.transfer-file`
   - There is only one service version: `1.0.0.0`

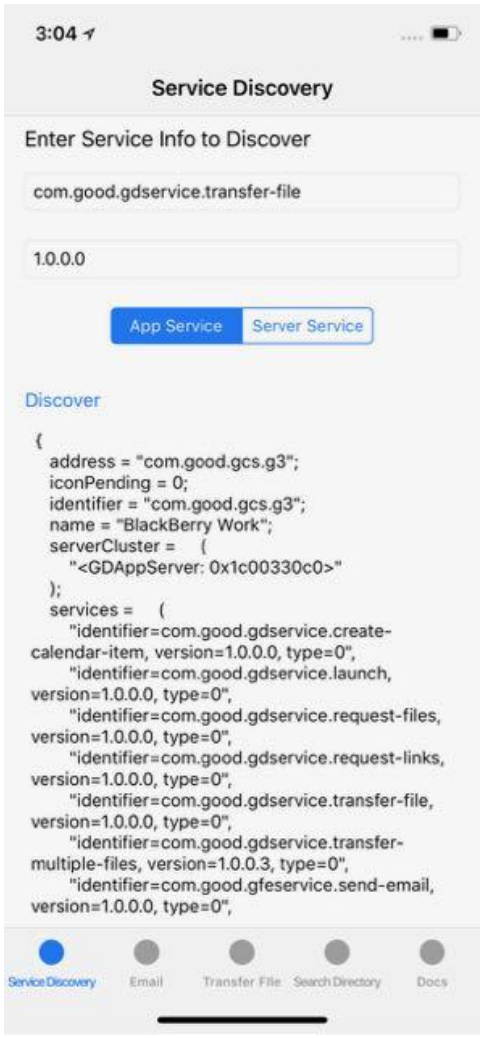
2. Install and activate a service provider application.

   Install and activate BlackBerry Work or AppKinetics sample using the BlackBerry UEM. Those apps provide the service Transfer File. To do this follow Appendix A. After BlackBerry Dynamics activation BlackBerry Work requires Active Sync setup. The password required is AD password used to login to BlackBerry UEM – DevUser!2018

   The application must be activated for the **same end user** in the same deployment as the application in which you code the following steps.

3. Now with AppKinetics and/or BlackBerry Work activated as well as the Services sample code it is time to retry Service Discovery.
   -Launch the Service app,
   -Select the Service Discovery tab,
   -Enter the Service ID and version as "`com.good.gdservice.transfer-file`" and "`1.0.0.0`"
   -Press **Discover** button

   You should see a similar result than below screenshot



4. If you have activated BlackBerry Work you can also check Service Discovery for the Send Email Service. As a reminder Service definitions can be found here - https://apps.good.com/#/services

5. The key API involved is [[GDiOS sharedInstance] getServiceProvidersFor:serviceID andVersion:serviceVersion andServiceType:type], usage of this API can be found in DiscoverServiceVC.m.

6. One key piece of information returned is the Provider application native address. This is required to communicate with the Service Provider.

By setting breakpoints in the code ensure that Service Discovery concept and API usage and returned parameters are well understood.

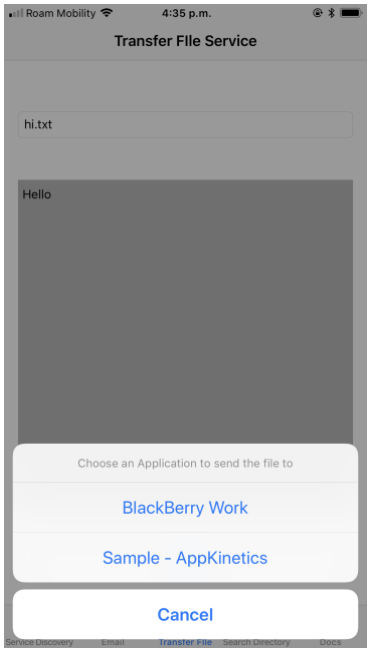
4.2. Exercise Two – Part 1- Consume a service – Transfer File Service

After Service Discovery returns that single/multiple Service Providers of the desired Service exist on the device it is now possible to consume that service. This part of the exercise consumes the Transfer File Service.

1. Navigate to the Transfer File tab in the Services Example. Enter a filename and file content into the UI and press 'Send' button

   At this point the following happens –

   - File is created with the desired name and content in the BlackBerry Dynamics Secure Container. Only files in Secure Container can be sent via App Kinetics
   - Service Discovery is executed to find providers of Transfer File Service.
   - Transfer is made to Service Provider



2. An important point highlight by the above screenshot. If you have more than one Service Provider for requested service activated on your device then
   [[GDiOS sharedInstance] getServiceProvidersFor:@"com.good.gdservice.transfer-file" andVersion:@"1.0.0.0" andServiceType:GDServiceTypeApplication];
   will return a ServiceProvider object for each one.

   In this case both BlackBerry Work and App Kinetics provide the Transfer File Service so are both returned. It is the consumer application's responsibility to make the choice on which provider to use. In the case of Services sample code, you can change programmatically which app to send the file to.

If only App Kinetics is activated, then the transfer is automatically completed without further user interaction.

3. The key class for consumption of Transfer File Service in Services example code is **TransferFileVC.m**.

   This uses the items defined in the Service Definition (Reminder Service definitions can be found online at - https://apps.good.com/#/services)

   The key method for Service consumption is
   [GDServiceClient sendTo:[serP address] withService:@"com.good.gdservice.transfer-file" withVersion:@"1.0.0.0" withMethod:@"transferFile" withParams:NULL withAttachments:files bringServiceToFront:GDEPreferPeerInForeground requestID:nil error:&err];
   this is called from the same file with correct parameters based on the Service Definition and the file to send.

   Make sure the parameters and call flow are understood by placing breakpoints in the code or asking questions. The same pattern can be used for consumption of any App Kinetics Service.

4.3. Exercise Two – Part 2 - Consume a service – Send Email Service

Note: This exercise can only be carried out if using a device with BlackBerry Work activated

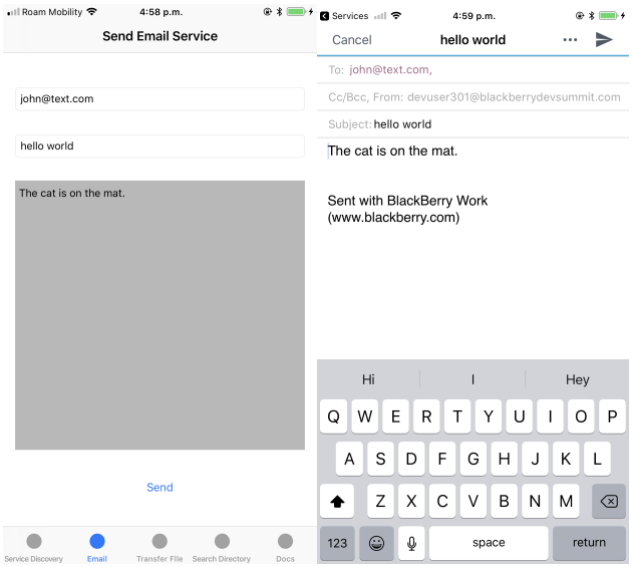
The intention of this exercise is to consume the Send Email Service which BlackBerry Work provides, and thus to show from a coding and structure perspective that consumption of any App Kinetics service follows the same pattern.

1. Navigate to the Email tab in the Services Example. Enter a TO email address, and email subject, and an email body and press SEND button.

   At this point the following happens –

   - Entered email parameters are packaged into object as per the Send Email Definition.
   - ServiceDiscovery is used to find the Provider of the Send Email Service. In this case it will be BlackBerry Work.
   - GDServiceClient.sendTo( ) is used to consume service by sending request to Service Provider.

   This flow is shown in below screenshots



2. The key class for consumption of this service in Services example code is **EmailServiceVC.m**.

   Every service request is assigned an ID, these IDs are then used to associate the request with success or failure callbacks. A successful request will cause UI code to be reset. A failure callback will cause the failure to be displayed to the user.

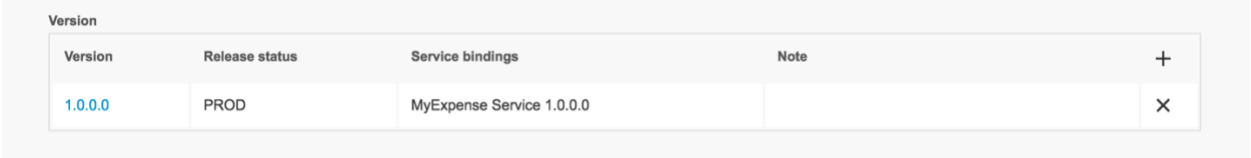
   Make sure the call flow and key methods are understood by placing breakpoints in the code or asking questions.

4.4. Exercise Three: Provide a custom service

The purpose of this exercise is to demonstrate an example of Providing a Service which other applications can consume.

The first stage is to bind Service to BlackBerry Dynamics application.

Note: This has already been done with Services Example in the BlackBerry Developer Summit UEM.



However, to implement your application as a Service Provider for another service you will need to bind that Service.

A Service consumer can send a Provider a request at any time. Service Provider needs to register a Listener to be able to handle this Service request.

The Service Definition of MyExpenses Service can be found in this workbook Appendix B.

The functionality provided by this Service –

- Services has a method where Consumer can query Expense limit which is returned by Provider.
- Service has a method where the Consumer can send an Expense item which the Provider will receive and store.
- Expense limit is set as a secure container.
- Received Expenses are stored in secure File.

There is no sample code supplied for the example which uses consumes the MyExpenses Service. Creation of this can be considered an 'extra credit' exercise and could reuse components of the Services sample code or an existing Blackberry Dynamics you have already created could be configured to be a consumer of this service.

5. BEMS Services Exercises

These exercises correspond to the: Consume a BEMS Service session.

These exercises are based on the same sample code, the Services sample application

5.1. Exercise One – Prepare BEMS Service usage

If the initial AppKinetics exercises were completed then the Services application will already be build / installed / activated on device or simulator. If not do that first.

Server Based Services are ones which are provided by a Server and not an application. However these Services are discovered in the same way as application services are discovered.

[[GDiOS sharedInstance] getServiceProvidersFor:@"" andVersion:@"" andServiceType:GDServiceTypeApplication]; is the key method to use, and the Server Services definitions can be found in list of public services - https://apps.good.com/#/services

In the Services application navigate to Service Discovery tab. Modify the code to discover the Docs and Directory Server Services (instead of the Transfer File and Send Email application application services which are used). Start by looking at DiscoverServiceVC.m

TIP – In addition to the Service ID and version remember to change the Service Provider type.

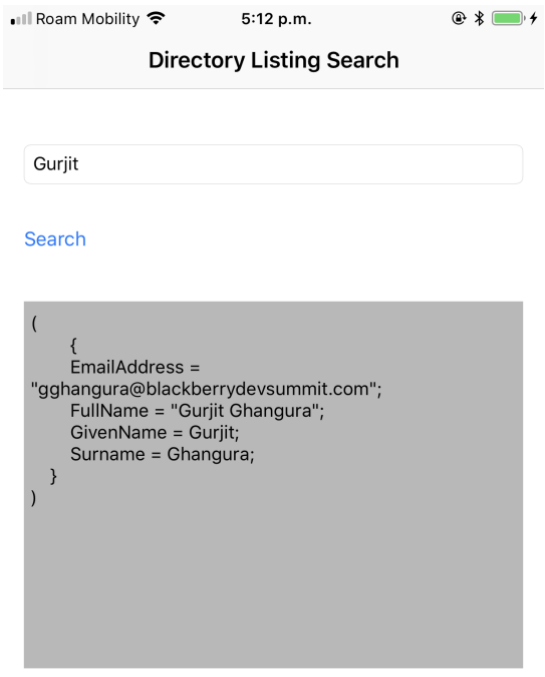
The important points from Server Service perspective are; Server Host, Server Port, Server Priority. This information is then used as basis to consume Server Based Services.

Make sure you understand the returned items from Service Discovery query by browsing code, setting breakpoints and asking questions.

5.2. Exercise Two – Consume Directory Service

The Directory Service provides a REST API to query & retrieve information from Active Directory. The specifics of the API can be found by reading API documentation which can be found via Service Definition -https://apps.good.com/#/services

Navigate to the Search Directory tab of the Services application. Enter a search query and select SEARCH. For example you could search for 'DevUser'. Below screenshot shows example output (A production environment AD would likely have more fields) –



The code which handles Service use is **DirectoryServicesVC.m**.

This code works as follows –

- On pressing SEARCH Service Discovery is used to find the BEMS Server host and port. These are then used together with the lookup API endpoint address to create a full query URL
- GDAuthToken is used to authenticate the User to BEMS, as such app code checks if it has a currently valid Auth Token for this URL. If it does not it requests one – GDUtility.getGDAuthToken()
- When GDAuthToken is successfully returned an AsyncTask is created and executed which makes the actual networking request.
- GD HTTP API is used as BEMS Service is an Enterprise Service and as such secure behind enterprise firewall.
- A query is formed into post the body which is sent. The response is parsed into a String and updated into the UI.

The core flow here is important, and the same for other Server Based Services –

1. Discover Server/Port for requested Server Service.
2. Form URL based on this and desired API endpoint (from Service definition).
3. Request GDAuthToken for URL (if don't already have one).
4. Make HTTP Request providing query.
5. Receive and process the response.

Make sure you understand this flow and code involved in **DirectoryServices.m** by adding breakpoints, changing query, asking questions.

5.3. Exercise Three – Consume Docs Service

Consume Docs Service is another provided example of consuming Server Based Service. In the Services Example application navigate to the Docs Listing Tab and hit "SEND LIST DOCS REQUEST". This will make a HTTP request to return contents of root of the Developer Summit docs site.

This code can be found in DocsServiceConsumer. It follows the same flow as previous example however shows use of different Service.

Make sure you understand the code involved by adding breakpoints, changing the query and asking questions.

5.4. Exercise Four – Auto Discover Service

This exercise is ‘extra credit’ no sample code is provided.

However existing code could be repurposed to make use of this Service. This is an example of making use of BEMS for auto discovery

REQUEST –

POST /api/autodiscover

```
{
  "Settings": [
    "EwsUrl",
    "EasUrl"
  ]
}
```

RESPONSE

```
{
"EasUrl":{
"Status": "ok",
"Url": "https://XXXX/Microsoft-Server-ActiveSync"
},
"EwsUrl":{
```

"Status": "ok",

"InternalUrl": "https://XXXX/EWS/Exchange.asmx",

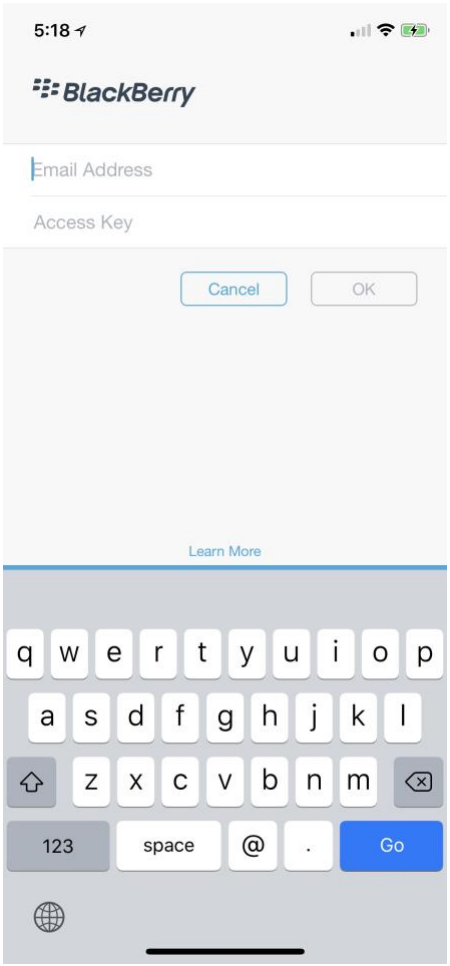
"ExternalUrl": "https://XXXX/EWS/Exchange.asmx"

}

}

Appendix A – Activate an Application in the BlackBerry Developer Lab

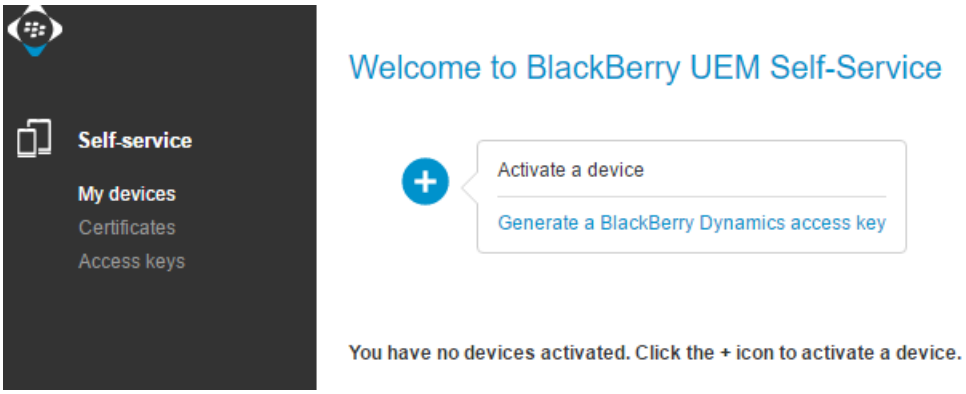
1. In activation screen of your sample application enter an email address of DevUser###@BlackBerryDevSummit.com, replacing ### with the number assigned to you when you arrived at the summit. The activation screen will appear as shown in the screenshot below.



2. Navigate to the BlackBerry UEM Self Service Portal to generate an access key: https://uem.blackberrydevsummit.com:10443/index.jsp

3. Log in using the following credentials:
   - Username: DevUser###
     - Replace ### with the number assigned to you when you arrived at the summit.
   - Password: DevUser!2018
   - Domain: BlackBerryDevSummit.com

4. Once logged in, click on the + button and choose “Generate a BlackBerry Dynamics access key”.



5. Return to your sample application and enter the access key you just generated.

6. Press the OK button.

7. The activation processing carousel will scroll through its stages.

8. Create a password that will be required to start the application.

The application user interface will then be displayed. Note that activation isn't necessary when you reinstall the application as an upgrade, without uninstalling.

Appendix B – MyExpenses Service Definition

MyExpenses is a private Service which the Services sample code is configured to implement Service Provider. Below is the Service Definition for this Service.

{

"title": "My Expense",

"service-id": "com.blackberrydevsummit.2017.services.my-expense",

"version": "1.0.0.0",

"description": "This services offers services of providing the expense limit to a requesting application and allow an application to set a expense item",

"methods": {

"getExpenseLimit": {

"description": "The consumer should specify itself to be brought to the foreground when sending a request with this method. The service provider should respond returning the expense limit (Integer) as parameter",

"params": {

"type": "null"

},

"result-value": {

"type": "integer",

"description": "Integer value of expense limit"

}

},

"setExpenseItem": {

"description": "",

"params": {

"type": "object",

"properties": {

"item": {

"description": "Name of expense item",

"type": "string"

},

"amount": {

```
                                    "description": "Amount of the expense item",
                                    "type": "integer
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}
```