



IBM MobileFirst Platform Foundation

---

# IBM MobileFirst™ Platform Foundation V8.0.0

**C# client-side API for native Windows 10 UWP, Windows 8  
Universal and Windows Phone 8.1 Universal apps**

## Note

Before using this information and the product it supports, read the information in “Notices” on page [Appendix A. Notices](#).

This edition applies to version V8.0.0 of IBM MobileFirst Platform Foundation and to all subsequent modifications and fix-packs, until otherwise indicated in new editions. This edition was updated last on 31 March 2016.

© Copyright IBM Corporation 2014, 2016

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

## About IBM®

See the [About IBM](http://www.ibm.com/ibm/us/en/) page (www.ibm.com/ibm/us/en/).

# Contents

<b>1</b>	<b>Worklight namespace overview .....</b>	<b>1</b>
<b>2</b>	<b>API reference .....</b>	<b>5</b>
2.1	Example Code.....	5
2.1.1	Example: connecting to the MobileFirst Server and calling a procedure .....	5
2.2	GatewayChallengeHandler Class .....	8
2.2.1	Method GetChallengeAnswer .....	8
2.2.2	Method GetLoginFormParameters .....	8
2.2.3	Method GetRealm.....	9
2.2.4	Method HandleChallenge .....	9
2.2.5	Method canHandleResponse.....	10
2.2.6	Method OnFailure .....	10
2.2.7	Method OnSuccess.....	11
2.2.8	Method ShouldSubmitChallengeAnswer .....	11
2.2.9	Method ShouldSubmitLoginForm .....	12
2.2.10	Method ShouldSubmitSuccess .....	12
2.2.11	Method ShouldCancel.....	13
2.3	IWorklightClient Interface .....	13
2.3.1	Property AuthorizationManager .....	13
2.3.2	Property HeartBeatInterval .....	14
2.3.3	Method Connect [Deprecated] .....	14
2.3.4	Method Connect (WorklightRequestOptions) [Deprecated].....	14

2.3.5	Method AddGlobalHeader .....	15
2.3.6	Method GetDeviceDisplayName() .....	16
2.3.7	Method InvokeProcedure(WorklightProcedureInvocationData).....	16
2.3.8	Method InvokeProcedure(WorklightProcedureInvocationData, Int32).....	17
2.3.9	Method RegisterChallengeHandler(GatewayChallengeHandler) .....	18
2.3.10	Method RegisterChallengeHandler(SecurityCheckChallengeHandler) .....	18
2.3.11	Method RemoveGlobalHeader .....	19
2.3.12	Method ResourceRequest(Uri, String, Int32).....	19
2.3.13	Method ResourceRequest(Uri, String, String) .....	20
2.3.14	Method SetDeviceDisplayName(String) .....	21
2.4	LoginFormInfo Class .....	22
2.4.1	Constructor LoginFormInfo .....	22
2.4.2	Property RequestHeaders .....	23
2.4.3	Property RequestMethod .....	23
2.4.4	Property RequestParameters .....	24
2.4.5	Property TimeoutInMs.....	24
2.4.6	Property Url.....	24
2.5	SecurityCheckChallengeHandler Class .....	24
2.5.1	Property SecurityCheck .....	25
2.5.2	Method GetChallengeAnswer .....	25
2.5.3	Method HandleChallenge .....	25
2.5.4	Method HandleFailure.....	26
2.5.5	Method HandleSuccess .....	26

2.5.6	Method ShouldCancel.....	27
2.5.7	Method ShouldSubmitChallengeAnswer .....	27
2.5.8	Method SubmitChallengeAnswer.....	28
2.6	WorklightAuthorizationManager Class .....	28
2.6.1	Property AuthorizationServerUri .....	29
2.6.2	Method ClearAccessToken.....	29
2.6.3	Method GetResourceScope.....	29
2.6.4	Method IsAuthorizationRequired(HttpWebResponse).....	30
2.6.5	Method IsAuthorizationRequired(HttpStatusCode, String) .....	31
2.6.6	Method Login .....	31
2.6.7	Method Logout.....	32
2.6.8	Method ObtainAccessToken.....	33
2.7	WorklightClient Class .....	33
2.7.1	Property AuthorizationManager .....	34
2.7.2	Property HeartBeatInterval .....	34
2.7.3	Method CreateInstance.....	34
2.7.4	Method Connect.....	35
2.7.5	Method Connect (WorklightRequestOptions) .....	35
2.7.6	Method AddGlobalHeader .....	36
2.7.7	Method GetDeviceDisplayName() .....	36
2.7.8	Method InvokeProcedure(WorklightProcedureInvocationData).....	37
2.7.9	Method InvokeProcedure(WorklightProcedureInvocationData, Int32).....	37
2.7.10	Method RegisterChallengeHandler(GatewayChallengeHandler) .....	38

2.7.11	Method RegisterChallengeHandler(SecurityChallengeHandler).....	39
2.7.12	Method RemoveGlobalHeader .....	39
2.7.13	Method ResourceRequest(Uri, String, Int32).....	40
2.7.14	Method ResourceRequest(Uri, String, String) .....	40
2.7.15	Method SetDeviceDisplayName(String) .....	41
2.8	WorklightProcedureInvocationData Class.....	42
2.8.1	Constructor WorklightProcedureInvocationData.....	42
2.8.2	Property AdapterName .....	43
2.8.3	Property CompressResponse.....	43
2.8.4	Property Parameters.....	44
2.8.5	Property ProcedureName .....	44
2.8.6	Method getInvocationDataMap .....	44
2.9	WorklightRequestOptions Class .....	44
2.9.1	Constructor WorklightRequestOptions.....	45
2.9.2	Property InvocationContext.....	45
2.9.3	Property TimeOut.....	46
2.10	WorklightResourceRequest Class .....	46
2.10.1	Property AllHeaders.....	47
2.10.2	Property HeaderNames .....	47
2.10.3	Property QueryParameters .....	47
2.10.4	Property Timeout .....	48
2.10.5	Property URL .....	48
2.10.6	Method AddHeader.....	48

2.10.7 Method GetFirstHeader .....	49
2.10.8 Method GetHeaders.....	49
2.10.9 Method RemoveHeaders .....	50
2.10.10 Method Send.....	50
2.10.11 Method Send(JObject) .....	51
2.10.12 Method Send(Byte[]) .....	51
2.10.13 Method Send(Dictionary(String, String)) .....	52
2.10.14 Method Send(String).....	53
2.10.15 Method SetHeader .....	53
2.10.16 Method SetHeaders .....	54
2.10.17 Method SetQueryParam .....	54
2.11 WorklightResponse Class .....	55
2.11.1 Property HTTPStatus.....	55
2.11.2 Property Message.....	55
2.11.3 Property ResponseJSON.....	56
2.11.4 Property ResponseText .....	56
2.11.5 Property Success.....	56



## About this document

This document is intended for developers who want to access IBM MobileFirst™ Platform Foundation services from Windows 8 Universal, Windows 10 UWP or Windows Phone 8.1 Universal applications that are written in C#. The document guides you through the available classes and methods.

## 1 Worklight namespace overview

This document contains APIs that are part of the `Worklight` namespace.

APIs that are part of the `IBM.Worklight` namespace are obsolete. For more information about deprecated APIs, see the table Windows C# API elements deprecated in V8.0.0 in the topic “[Deprecated features and API elements](#)”

([https://www.ibm.com/support/knowledgecenter/SSHS8R\\_8.0.0/com.ibm.worklight.getstart.doc/what\\_s\\_new/c\\_deprecations.html](https://www.ibm.com/support/knowledgecenter/SSHS8R_8.0.0/com.ibm.worklight.getstart.doc/what_s_new/c_deprecations.html)).

The IBM MobileFirst Platform Foundation C# client-side API for native Windows 8 Universal, Windows 10 UWP and Windows Phone 8.1 Universal applications exposes two main capabilities:

- Calling back-end services to retrieve data and perform back-end transactions.
- Writing custom log lines for reporting and auditing purposes.

The IBM MobileFirst Platform Foundation C# client-side API for native Windows 8 Universal, Windows 10 UWP and Windows Phone 8.1 Universal applications is available as part of the IBM MobileFirst Platform Studio.

Type	Name	Description	Implemented By
Class	<code>GatewayChallengeHandler</code>	This is a base class you must implement to create your own custom Challenge Handlers. The custom Challenge Handler logic required to authenticate against a realm defined on the server must be written in your implementation.	IBM
Class	<code>LoginFormInfo</code>	This class that contains properties required to submit a login form to the server in response to a challenge on a realm protected by a <code>FormBasedAuthenticator</code> . This is similar to the <code>submitLoginForm</code> method in Worklight Native API.	IBM

Type	Name	Description	Implemented By
Class	SecurityCheckChallengeHandler	Abstract base class for the IBM MobileFirst Platform Challenge Handlers. You must extend it to implement your own version of an IBM MobileFirst Platform Challenge Handler, for example RemoteDisableChallengeHandler.	IBM
Class	WorklightAuthorizationManager	This class manages the entire OAuth flow, from client registration to token generation.	IBM
Class	WorklightClient	This class implements the IWorklightClient interface	Application developer
Class	WorklightNotificationEventArgs	Worklight notification event arguments.	IBM
Class	WorklightProcedureInvocationData	This class contains all necessary data to call a procedure.	IBM
Class	WorklightRequestOptions		IBM

Type	Name	Description	Implemented By
Class	WorklightResourceRequest	<p>This class encapsulates a resource request. The resource may be an adapter on the MobileFirst Server, or an external resource. The class provides several 'send' methods, with different inputs for the body of a request. In addition, the 'send' methods support two types of response listener: <code>WLResponseListener</code> - The <code>onSuccess</code> method of this listener is called and provided with an instance of the <code>WLResponse</code> class. The content of the response is be read into the <code>WLResponse</code> instance by the platform, and will be accessible through methods of <code>WLResponse</code>. In case of a failure, the <code>onFailure</code> method of the listener is called and provided with an instance of the <code>WLFailResponse</code> class, that will contain all the information about the failure.</p> <p><code>WLHttpResponseListener</code> - The <code>onSuccess</code> method of this listener is called and provided with the original HTTP response object that was received from the server. The platform does not attempt to read or parse the response in any way. In case of a failure the <code>onFailure</code> method is called and provided with either the response from the server if one was received, or the exception that was thrown during the execution of this request.</p> <p>Regardless of what type of listener was used, a successful response is any response with a status in the 2xx range. These responses are delivered to the <code>onSuccess</code> method. A response with a 4xx or 5xx status is considered a failure, and is delivered to the <code>onFailure</code> method.</p>	IBM

Type	Name	Description	Implemented By
Class	<code>WorklightResponse</code>	This class contains the result of a procedure invocation.	IBM
Interface	<code>IWorklightClient</code>	This interface exposes methods that you use to communicate with the Worklight Server.  You must get a specific instance using <code>WorklightClient.CreateInstance()</code>	IBM

*Table 1-1: IBM MobileFirst Platform Foundation C# client-side API for Windows 8 Universal, Windows 10 UWP and Windows Phone 8.1 Universal applications – packages, classes, interfaces, and files*

## 2 API reference

### 2.1 Example Code

The following code samples show how to use the IBM MobileFirst Platform Foundation C# client-side API for native Windows 8 Universal, Windows 10 UWP and Windows Phone 8.1 Universal applications.

#### 2.1.1 Example: connecting to the MobileFirst Server and calling a procedure

##### Initializing the MobileFirst Client

```
Worklight.IWorklightClient _newClient =  
Worklight.WorklightClient.CreateInstance();
```

##### Registering the challengehandler

```
_newClient.RegisterChallengeHandler(new  
DummySecurityChallengeHandler("securitycheck"));
```

##### Adapter Invocation (ResourceRequest)

```
Uri url = new Uri("/adapters/adaptername/adapterprocedure",  
UriKind.Relative);  
  
WorklightResourceRequest r2 = _newClient.ResourceRequest(url,  
"Method", "securitycheck");  
  
WorklightResponse res = await r2.Send();
```

##### Implementing the GatewayChallengeHandler

```
public class DummySecurityChallengeHandler :
Worklight.GatewayChallengeHandler
{
    public JObject LoginFormParameters { get; set; }
    private bool authSuccess = false;
    private bool isAdapterAuth = false;
    public string Realm;
    private bool shouldsubmitchallenge = false;
    public DummySecurityChallengeHandler(string realm)
    {
        this.Realm = realm;
    }
    public override JObject GetChallengeAnswer()
    {
        return LoginFormParameters;
    }

    public override string GetRealm()
    {
        return Realm;
    }

    public override bool ShouldSubmitSuccess()
    {
        return authSuccess;
    }

    public override bool ShouldSubmitChallengeAnswer()
    {
        return shouldsubmitchallenge;
    }

    public void submitResponse(string user, string password)
    {
        JObject answer = new JObject();
        answer.Add("user",user);
    }
}
```

```
        answer.Add("password",password);
        shouldsubmitchallenge = true;
    }
    public override void HandleChallenge(WorklightResponse
challenge)
    {

        LoginFormParameters = new JObject();

        LoginFormParameters.Add("user", "user");

        LoginFormParameters.Add("password", "user");

        shouldsubmitchallenge = true;
    }

    public override bool canHandleResponse(WorklightResponse
response)
    {
        Debug.WriteLine("Determining if its a custom response");
        if (response == null || response.ResponseText == null ||
!(response.ResponseText.Contains("j_security_check")))
        {
            return false;
        }

        return true;
    }

    public override void OnSuccess(WorklightResponse success)
    {

        Debug.WriteLine(success.ToString());

    }
    public override void OnFailure(WorklightResponse failure)
    {
```



```
        Debug.WriteLine(failure.ToString());  
    }  
}
```

## 2.2 GatewayChallengeHandler Class

### Syntax

```
public abstract class GatewayChallengeHandler
```

### Description

This is a base class you must implement to create your own custom GatewayChallengeHandlers. The custom GatewayChallengeHandler logic required to authenticate against a realm defined on the server must be written in your implementation.

The GatewayChallengeHandler type exposes the following members.

### 2.2.1 Method GetChallengeAnswer

#### Syntax

```
public abstract JObject GetChallengeAnswer()
```

#### Description

Override this method to return the challenge answer that must be returned to the server.

#### Return Value

Type: JObject

This method must return the challenge answer that should be sent to the server.

### 2.2.2 Method GetLoginFormParameters

#### Syntax

```
public virtual LoginFormInfo GetLoginFormParameters()
```

## Description

Override this if you want to provide an `LoginFormInfo` object with all the parameters required to submit a form in response to a challenge on a realm protected by an `FormBasedAuthenticator`.

## Return Value

Type: `LoginFormInfo`

Default: `null`

### 2.2.3 Method `GetRealm`

#### Syntax

```
public abstract string GetRealm()
```

## Description

The realm defined on the server that your custom Challenge Handler will deal with.

## Return Value

Type: `String`

realm

### 2.2.4 Method `HandleChallenge`

#### Syntax

```
public abstract void HandleChallenge(  
    WorklightResponse challenge  
)
```

## Description

This method is called whenever `canHandleResponse(WorklightResponse)` returns a true value. You must implement this method to handle the challenge logic, for example to display the login screen. You can handle this for a particular security realm.

## Parameters

Type	Name	Description
<code>Worklight.WorklightResponse</code>	challenge	The challenge that is presented by the server.

Table 2-1: Method `HandleChallenge` parameters

## 2.2.5 Method `canHandleResponse`

### Syntax

```
public virtual bool canHandleResponse(
    WorklightResponse response
)
```

### Description

Override this method if you want to decide whether `HandleChallenge(WorklightResponse)` will be called or not. Here you can parse the response from the Worklight server to determine whether or not your custom `GatewayChallengeHandler` will handle the challenge. Worklight will then call the `HandleChallenge` method depending on the return value. For example, in a realm protected by a `AdapterAuthenticator`, the response from the Worklight server might contain a JSON value of `authRequired : true`.

### Parameters

Type	Name	Description
<code>Worklight.WorklightResponse</code>	response	The response from the Worklight server which your implementation must parse

Table 2-2: Method `canHandleResponse` parameters

### Return Value

Type: Boolean

true if the response is meant for your Challenge Handler, false otherwise. Default:true

## 2.2.6 Method `OnFailure`

### Syntax

```
public abstract void OnFailure(  
    WorklightResponse response  
)
```

### Description

Is called by the framework in case of a failure.

### Parameters

Type	Name	Description
<code>Worklight.WorklightResponse</code>	response	The response from the Worklight server which your implementation must parse.

Table 2-3: Method `OnFailure` parameters

## 2.2.7 Method `OnSuccess`

### Syntax

```
public abstract void OnSuccess(  
    WorklightResponse challenge  
)
```

### Description

Is called by the framework in case of a success.

### Parameters

Type	Name	Description
<code>Worklight.WorklightResponse</code>	challenge	The challenge that is presented by the server.

Table 2-4: Method `OnSuccess` parameters

## 2.2.8 Method `ShouldSubmitChallengeAnswer`

### Syntax

```
public abstract bool ShouldSubmitChallengeAnswer()
```

### Description

Override this method to return whether or not the MobileFirst SDK should submit a challenge answer to the server. Override the `GetChallengeAnswer` method such that it returns the challenge answer.

### Return value

Type: Boolean

true if the SDK must return a challenge answer to the server.

## 2.2.9 Method `ShouldSubmitLoginForm`

### Syntax

```
public virtual bool ShouldSubmitLoginForm()
```

### Description

Override this if you want to set this property to true in your implementation if the realm is protected by an `FormBasedAuthenticator`. When this property is set to true, Worklight will send a response back to the server similar to submitting a form. The `LoginFormParameters` property must be populated with the necessary information to submit the form. Default: false

### Return value

Type: Boolean

Default: false

## 2.2.10 Method `ShouldSubmitSuccess`

### Syntax

```
public abstract bool ShouldSubmitSuccess()
```

### Description

Indicates if the auth attempt successful.

### Return value

Type: Boolean

true, If the challenge handler deems the authentication a success, return a true, false otherwise.

## 2.2.11 Method ShouldCancel

### Syntax

```
public virtual bool ShouldCancel()
```

### Description

Override this if you want to return a failure and end the authentication process. When this method returns true, MobileFirst Platform understands that the challenge was unsuccessful and that you no longer want to take any actions to attempt to resolve the problem. This method returns control to MobileFirst Platform for further handling. For example, call this method only when you know that several authentication attempts were unsuccessful and you do not want the user to continue attempting to authenticate into the security check. When this method returns true, you should override the `GetFailureResponse` method to return a `WorklightResponse` along with the failure.

### Return value

Type: Boolean

**true**, in case of in case of cancelling a challenge handler; **false** otherwise

## 2.3 IWorklightClient Interface

### Syntax

```
public interface IWorklightClient
```

### Description

This interface exposes methods that you use to communicate with the Worklight Server.

### Remarks

You must get a specific instance using `WorklightClient.CreateInstance()`

### 2.3.1 Property AuthorizationManager

#### Syntax

```
WorklightAuthorizationManager AuthorizationManager { get; }
```

#### Description

Provides a handle to the Authorization Manager

### Property value

Type: WorklightAuthorizationManager

The WorklightAuthorizationManager object.

### 2.3.2 Property HeartBeatInterval

#### Syntax

```
int HeartBeatInterval { get; set; }
```

#### Description

Interval, in seconds, at which the Worklight Server sends the heartbeat signal. You use the heartbeat signal to ensure that the session with the server is kept alive when the app does not issue any call to the server, such as invokeProcedure. By default, the interval is set to 20 minutes.

### Property value

Type: Int32

### 2.3.3 Method Connect [Deprecated]

#### Syntax

```
Task<WorklightResponse> Connect ()
```

#### Description

This method sends an initialization request to the MobileFirst Platform Server, establishes a connection with the server, and validates the application version. This method is deprecated, use `WorklightAuthorizationManager.ObtainAccessToken(string scope);` instead.

#### Return Value

Type: Task<WorklightResponse>

returns WorklightResponse object.

### 2.3.4 Method Connect (WorklightRequestOptions) [Deprecated]

#### Syntax

```
Task<WorklightResponse> Connect (  
  
    WorklightRequestOptions requestOptions  
  
    )
```

### Description

This method sends an initialization request to the MobileFirst Platform Server, establishes a connection with the server, and validates the application version. This method is deprecated, use `WorklightAuthorizationManager.ObtainAccessToken(string scope);` instead.

### Parameters

Type	Name	Description
WorklightRequestOptions	requestOptions	WorklightRequestOptions instance

Table 2-5: Method `Connect(WorklightRequestOptions)` parameters

### Return Value

Type: `Task<WorklightResponse>`

returns `WorklightResponse` object.

## 2.3.5 Method AddGlobalHeader

### Syntax

```
void AddGlobalHeader(  
  
    string headerkey,  
  
    string headerValue  
  
    )
```

### Description

You use this method to add a global header, which is sent on each request.



## Parameters

Type	Name	Description
System.String	headerKey	Name of the header.
System.String	headerValue	Value of the header.

Table 2-6: Method `AddGlobalHeader(string, string)` parameters

### 2.3.6 Method `GetDeviceDisplayName()`

#### Syntax

```
Task<WorklightDeviceDisplayName> GetDeviceDisplayName()
```

#### Description

Gets Friendly Device `DisplayName` for the client in the server. It accepts variable `DeviceDisplayName` of type string which you want to set.

#### Return Value

Type: `Task(WorklightDeviceDisplayName)`

returns `WorklightDeviceDisplayName` object.

### 2.3.7 Method `InvokeProcedure(WorklightProcedureInvocationData)`

#### Syntax

```
Task<WorklightResponse> InvokeProcedure (  
    WorklightProcedureInvocationData data  
)
```

#### Description

This method sends an asynchronous call to an adapter procedure. The return status and the return data is available in the `WorklightResponse` object returned.

#### Parameters

Type	Name	Description
Worklight.Worklight ProcedureInvocation Data	data	The invocation procedure data parameters to send on the request.

Table 2-7: Method `InvokeProcedure(WorklightProcedureInvocationData)` parameters

### Return Value

Type: `Task(WorklightResponse)`

The `WorklightResponse`.

Type: `Task(WorklightResponse)`

Returns the response from server

### 2.3.8 Method `InvokeProcedure(WorklightProcedureInvocationData, Int32)`

#### Syntax

```
Task<WorklightResponse> InvokeProcedure (
    WorklightProcedureInvocationData data,
    int timeOut
)
```

#### Description

This method sends an asynchronous call to an adapter procedure. The return status and the return data is available in the `WorklightResponse` object returned.

#### Parameters

Type	Name	Description
Worklight.Worklight ProcedureInvocation Data	data	The invocation procedure data parameters to send on the request.
System.Int32	timeOut	Use to change the request timeout which is an integer value.

Table 2-8: Method `InvokeProcedure(WorklightProcedureInvocationData, Int32)` parameters**Return Value**

Type: `Task<WorklightResponse>`

The `WorklightResponse`.

Type: `Task<WorklightResponse>`

Returns the response from server

**2.3.9 Method `RegisterChallengeHandler(GatewayChallengeHandler)`****Syntax**

```
void RegisterChallengeHandler(
    GatewayChallengeHandler challengeHandler
)
```

**Description**

You can use this method to register a Challenge Handler in the client. You must use this method when you implement custom challenge handlers. **Important:** you must call this method at the beginning of your application after you initialize `WLClient`.

**Parameters**

Type	Name	Description
<code>Worklight.GatewayChallengeHandler</code>	<code>challengeHandler</code>	The custom challenge handler instance

Table 2-9: Method `RegisterChallengeHandler(GatewayChallengeHandler)` parameters**2.3.10 Method `RegisterChallengeHandler(SecurityCheckChallengeHandler)`****Syntax**

```
void RegisterChallengeHandler(
    SecurityCheckChallengeHandler worklightChallengeHandler
)
```

**Description**

You can use this method to register a Challenge Handler in the client. You must use this method when you implement custom challenge handlers, or when you customize the Remote Disable / Notify Challenge Handler.

### Parameters

Type	Name	Description
Worklight.SecurityCheckChallengeHandler	worklightChallengeHandler	The custom challenge handler instance

Table 2-10: Method RegisterChallengeHandler(SecurityChallengeHandler) parameters

### 2.3.11 Method RemoveGlobalHeader

#### Syntax

```
void RemoveGlobalHeader(
    string headerKey
)
```

#### Description

You use this method to remove a global header. Then, the header is no longer sent on each request.

### Parameters

Type	Name	Description
System.string	headerKey	Name of the header.

Table 2-11: Method RemoveGlobalHeader parameters

### 2.3.12 Method ResourceRequest(Uri, String, Int32)

#### Syntax

```
WorklightResourceRequest ResourceRequest (
    Uri uri,
    String method,
    int timeout
)
```

**Description**

Constructs a new resource request with the specified URL, using the specified HTTP method.

**Parameters**

Type	Name	Description
System.Uri	uri	The resource URL, it can be either relative or absolute.
System.String	method	The HTTP method to use.
System.Int32	timeout	Use to change request timeout which is an integer value.

Table 2-6: Method ResourceRequest(Uri, String, Int32) parameters

**Return Value**

Type: WorklightResourceRequest

The WorklightResourceRequest object.

**2.3.13 Method ResourceRequest(Uri, String, String)****Syntax**

```
WorklightResourceRequest ResourceRequest (
    Uri uri,
    String method,
    String scope
)
```

**Description**

Constructs a new resource request with the specified URL, using the specified HTTP method

**Parameters**

Type	Name	Description
System.Uri	uri	The resource URL, it can be either relative or absolute.
System.String	method	The HTTP method to use.
System.String	scope	A string representing the scope required to access the resource.

Table 2-13: Method ResourceRequest(Uri, String, String) parameters

**Return Value**

Type: WorklightResourceRequest

The WorklightResourceRequest object.

**2.3.14 Method SetDeviceDisplayName(String)****Syntax**

```
Task<WorklightDeviceDisplayName> SetDeviceDisplayName (
    string DeviceDisplayName
)
```

**Description**

Sets friendly device display name for the client in the server.

**Parameters**

Type	Name	Description
System.String	DeviceDisplay Name	The friendly device display name.

Table 2-7: Method SetDeviceDisplayName(String) parameters

**Return Value**

Type: WorklightDeviceDisplayName

returns WorklightDeviceDisplayName object contains Success or Failure Response.

## 2.4 LoginFormInfo Class

### Syntax

```
public class LoginFormInfo
```

### Description

This class contains properties required to submit a login form to the server in response to a challenge on a realm protected by a FormBasedAuthenticator. This is similar to the submitLoginForm method in Worklight Native API

### Inheritance Hierarchy

System.Object

Worklight.LoginFormInfo

### 2.4.1 Constructor LoginFormInfo

#### Syntax

```
public LoginFormInfo(  
    string url,  
    Dictionary<string, string> requestParameters,  
    Dictionary<string, string> requestHeaders,  
    int timeOutInMs,  
    string requestMethod  
)
```

#### Description

Instanciates the LoginFormInfo

#### Parameters

Type	Name	Description
System.String	url	url

Type	Name	Description
System.Collections.Generic.Dictionary(String, String)	requestParameters	Request parameters
System.Collections.Generic.Dictionary(String, String)	requestHeaders	Request headers
System.Int32	timeOutInMs	Request time out
System.String	requestMethod	Request method

Table 2-15: Constructor `LoginFormInfo` parameters

## 2.4.2 Property `RequestHeaders`

### Syntax

```
public Dictionary<string, string> RequestHeaders { get; set; }
```

### Description

A Dictionary object consisting of additional headers to be included in the request made while submitting the form.

### Property Value

Type: Dictionary(String, String)

## 2.4.3 Property `RequestMethod`

### Syntax

```
public string RequestMethod { get; set; }
```

### Description

The HTTP method that you must use. Acceptable values are GET, POST. The default value is POST.

### Property Value

Type: String



## 2.4.4 Property RequestParameters

### Syntax

```
public Dictionary<string, string> RequestParameters { get; set; }
```

### Description

A Dictionary object consisting of request parameters to be included in the request made while submitting the form.

### Property Value

Type: Dictionary(String, String)

## 2.4.5 Property TimeoutInMs

### Syntax

```
public int TimeoutInMs { get; set; }
```

### Description

The timeout value in milliseconds. The default is 10000 ms. To disable timeout, set this to 0.

### Property Value

Type: Int32

## 2.4.6 Property Url

### Syntax

```
public string Url { get; set; }
```

### Description

Absolute URL if the user sends an absolute url that starts with http:// or https:// Otherwise, URL relative to the Worklight Server

### Property Value

Type: String

## 2.5 SecurityCheckChallengeHandler Class

### Syntax

```
public abstract class SecurityCheckChallengeHandler
```

### Description

Abstract base class for the IBM MobileFirst Platform Challenge Handlers. You must extend it to implement your own version of an IBM MobileFirst Platform Challenge Handler, for example RemoteDisableChallengeHandler.

### Inheritance Hierarchy

System.Object

Worklight.SecurityCheckChallengeHandler

## 2.5.1 Property SecurityCheck

### Syntax

```
public abstract string SecurityCheck { get; set; }
```

### Description

The security check to which this Challenge Handler will respond to.

### Property Value

Type: String

## 2.5.2 Method GetChallengeAnswer

### Syntax

```
public abstract JObject GetChallengeAnswer()
```

### Description

This method must return the challenge answer that should be sent to the server.

### Return Value

Type: JObject

This method must return the challenge answer that should be sent to the server.

## 2.5.3 Method HandleChallenge

### Syntax

```
public abstract void HandleChallenge(  
  
    Object challenge  
  
)
```

### Description

This method is called whenever `[:canHandleResponse]` returns a true value. You must implement this method to handle the challenge logic, for example to display the login screen. You can handle this for a particular security realm.

### Parameters

Type	Name	Description
System.Object	challenge	The challenge that is presented by the server.

Table 2-8: Method `HandleChallenge` parameters

## 2.5.4 Method `HandleFailure`

### Syntax

```
public abstract void HandleFailure(  
  
    JObject error  
  
)
```

### Description

This method is called when the MobileFirst Platform Server reports an authentication failure.

### Parameters

Type	Name	Description
JObject	error	The details of the error.

Table 2-9: Method `HandleFailure` parameters

## 2.5.5 Method `HandleSuccess`

### Syntax

```
public abstract void HandleSuccess(  
  
    JObject identity  
  
)
```

### Description

This method is called when the MobileFirst Platform Server reports an authentication success.

### Parameters

Type	Name	Description
JObject	identity	

Table 2-10: Method HandleSuccess parameters

## 2.5.6 Method ShouldCancel

### Syntax

```
public virtual bool ShouldCancel()
```

### Description

Override this if you want to return a failure and end the authentication process. When this method returns true, MobileFirst Platform understands that the challenge was unsuccessful and that you no longer want to take any actions to attempt to resolve the problem. This method returns control to MobileFirst Platform for further handling. For example, call this method only when you know that several authentication attempts were unsuccessful and you do not want the user to continue attempting to authenticate into the security check.

### Return Value

Type: Boolean

**true**, in case of cancelling a challenge handler; **false** otherwise

## 2.5.7 Method ShouldSubmitChallengeAnswer

### Syntax

```
public abstract bool ShouldSubmitChallengeAnswer()
```

## Description

Override this method to return whether or not the MobileFirst SDK should submit a challenge answer to the server. Override the `GetChallengeAnswer` method such that it returns the challenge answer.

## Return Value

Type: Boolean

This method must return true if the SDK must return a challenge answer to the server.

## 2.5.8 Method `SubmitChallengeAnswer`

### Syntax

```
public abstract bool ShouldSubmitChallengeAnswer(  
    Object answer  
)
```

### Description

Send the answer back to the request.

### Parameters

Type	Name	Description
Object	answer	The answer to send back to the server.

Table 2-11: Method `SubmitChallengeAnswer(Object)` parameters

## 2.6 WorklightAuthorizationManager Class

### Syntax

```
public abstract class WorklightAuthorizationManager
```

### Description

This class manages the entire OAuth flow, from client registration to token generation.

### Inheritance Hierarchy

System.Object

Worklight.WorklightAuthorizationManager

The `WorklightAuthorizationManager` type exposes the following members.

### 2.6.1 Property `AuthorizationServerUri`

#### Syntax

```
public abstract Uri AuthorizationServerUri { get; set; }
```

#### Description

Gets or sets the `AuthorizationServerUri`.

#### Property Value

Type: `Uri`

### 2.6.2 Method `ClearAccessToken`

#### Syntax

```
public abstract void ClearAccessToken(  
    WorklightAccessToken accessToken  
)
```

#### Description

This method clears any previous access token.

#### Parameters

Type	Name	Description
<code>WorklightAccessToke n</code>	<code>accessToken</code>	The <code>accessToken</code> provided by the server to the client.

Table 2-20: Method `ClearAccessToken(WorklightAccessToken)` parameters

### 2.6.3 Method `GetResourceScope`

#### Syntax

```
public abstract string GetResourceScope(
```

```

    HttpResponseMessage response
)

```

### Description

Returns the scope that is required by the resource that produced the given response. This method expects to be given only response objects for which the method `IsAuthorizationRequired(HttpResponse)` returns true.

### Parameters

Type	Name	Description
String	response	HTTP response object

Table 2-21: Method `GetResourceScope(String)` parameters

### Return Value

Type: String

String representing the scope required by the resource.

## 2.6.4 Method `IsAuthorizationRequired(HttpWebResponse)`

### Syntax

```

public abstract bool IsAuthorizationRequired(
    HttpResponseMessage response
)

```

### Description

Checks whether the response is a MobileFirst Platform OAuth error.

### Parameters

Type	Name	Description
System.Net.HttpWebResponse	response	HTTP response object

Table 2-22: Method `IsAuthorizationRequired (HttpWebResponse)` parameters

## Return Value

Type: Boolean

true, if the response is a MobileFirst Platform OAuth error, false otherwise.

## 2.6.5 Method IsAuthorizationRequired(HttpStatusCode, String)

### Syntax

```
public abstract bool IsAuthorizationRequired(
    HttpStatusCode status,
    String authenticationHeader
)
```

### Description

Checks whether the response is a MobileFirst Platform OAuth error.

### Parameters

Type	Name	Description
System.Net.HttpStatusCode	status	HTTP status code
System.String	authenticationHeader	Value of the authentication header

Table 2-23: Method IsAuthorizationRequired (HttpWebResponse, String) parameters

## Return Value

Type: Boolean

true, if the response is a MobileFirst Platform OAuth error, false otherwise.

## 2.6.6 Method Login

### Syntax

```
public abstract Task<WorklightResponse> Login(
    String securityCheck,
    JObject credentials
)
```



## Description

Login to the specified security check.

## Parameters

Type	Name	Description
System.String	securityCheck	The securityCheck used to login.
JObject	credentials	The credentials used to login.

Table 2-24: Method Login(String, JObject) parameters

## Return Value

Type: Task(WorklightResponse)

Return WorklightResponse from the server which contains return status and response data.

## 2.6.7 Method Logout

### Syntax

```
public abstract Task<WorklightResponse> Logout (
    String securityCheck
)
```

## Description

Logout of the specified security check.

## Parameters

Type	Name	Description
System.String	securityCheck	The securityCheck used to perform logout operation.

Table 2-12: Method Logout(String) parameters

## Return Value

Type: Task(WorklightResponse)

Return WorklightResponse from the server which contains return status and response data.

## 2.6.8 Method ObtainAccessToken

### Syntax

```
public abstract Task<WorklightAccessToken> ObtainAccessToken(  
    String scope  
)
```

### Description

Explicit call to obtain the authorization header.

### Parameters

Type	Name	Description
System.String	scope	Scope required

Table 2-13: Method ObtainAccessToken(String) parameters

### Return Value

Type: Task(WorklightaccessToken)

A WorklightAccessToken object with the information about the access token for the given scope

## 2.7 WorklightClient Class

### Syntax

```
public class WorklightClient
```

### Description

This class implements the IWorklightClient interface

### Inheritance Hierarchy

System.Object

Worklight.WorklightClient

## 2.7.1 Property AuthorizationManager

### Syntax

```
public WorklightAuthorizationManager AuthorizationManager { get; }
```

### Description

Provides a handle to the Authorization Manager.

### Property Value

Type: WorklightAuthorizationManager

## 2.7.2 Property HeartBeatInterval

### Syntax

```
public int HeartBeatInterval { get; set; }
```

### Description

Interval, in seconds, at which the Worklight Server sends the heartbeat signal to ensure that the session with the server is kept alive when the app does not issue any call to the server, such as invokeprocedure. By default the interval is set to 20 minutes.

### Property Value

Type: Int32

## 2.7.3 Method CreateInstance

### Syntax

```
public static IWorklightClient CreateInstance()
```

### Description

You must get a specific instance using this method.

### Return Value

Type: IWorklightClient

Returns IWorklightClient reference.

## 2.7.4 Method Connect [Deprecated]

### Syntax

```
Task<WorklightResponse> Connect ()
```

### Description

This method sends an initialization request to the MobileFirst Platform Server, establishes a connection with the server, and validates the application version. This method is deprecated, use `WorklightAuthorizationManager.ObtainAccessToken(string scope);` instead.

### Return Value

Type: `Task<WorklightResponse>`

returns `WorklightResponse` object.

## 2.7.5 Method Connect (WorklightRequestOptions) [Deprecated]

### Syntax

```
Task<WorklightResponse> Connect (
    WorklightRequestOptions requestOptions
)
```

### Description

This method sends an initialization request to the MobileFirst Platform Server, establishes a connection with the server, and validates the application version. This method is deprecated, use `WorklightAuthorizationManager.ObtainAccessToken(string scope);` instead.

### Parameters

Type	Name	Description
<code>WorklightRequestOptions</code>	<code>requestOptions</code>	<code>WorklightRequestOptions</code> instance

Table 2-14: Method `Connect(WorklightRequestOptions)` parameters

## Return Value

Type: Task<WorklightResponse>

Returns WorklightResponse object.

## 2.7.6 Method AddGlobalHeader

### Syntax

```
public void AddGlobalHeader(  
    string headerkey,  
    string headerValue  
)
```

### Description

You use this method to add a global header, which is sent on each request.

### Parameters

Type	Name	Description
System.String	headerKey	Name of the header.
System.String	headerValue	Value of the header.

Table 2-15: Method AddGlobalHeader(String, String) parameters

## 2.7.7 Method GetDeviceDisplayName()

### Syntax

```
Task<WorklightDeviceDisplayName> GetDeviceDisplayName()
```

### Description

Gets Friendly Device DisplayName for the client in the server. It accepts variable DeviceDisplayName of type string which you want to set.

### Return Value

Type: Task(WorklightDeviceDisplayName)

returns WorklightDeviceDisplayName object.

## 2.7.8 Method InvokeProcedure(WorklightProcedureInvocationData)

### Syntax

```
public Task<WorklightResponse> InvokeProcedure (  
    WorklightProcedureInvocationData data  
)
```

### Description

This method sends an asynchronous call to an adapter procedure. The return status and the return data are available in the WorlightResponse object.

### Parameters

Type	Name	Description
Worklight.WorklightProcedureInvocationData	data	The ProcedureInvocationData parameters to send on the request.

Table 2-16: Method InvokeProcedure(WorklightProcedureInvocationData) parameters

### Return Value

Type: Task(WorklightResponse)

Returns the response from the server.

This method is **Obsolete**.

## 2.7.9 Method InvokeProcedure(WorklightProcedureInvocationData, Int32)

### Syntax

```
public Task<WorklightResponse> InvokeProcedure (  
    WorklightProcedureInvocationData data,  
    int timeout
```

)

## Description

This method sends an asynchronous call to an adapter procedure. The return status and the return data are available in the WorklightResponse object returned.

## Parameters

Type	Name	Description
Worklight.WorklightProcedureInvocationData	data	The ProcedureInvocationData parameters to send on the request.
System.Int32	timeout	The timeout in milliseconds for the request.

Table 2-30: Method InvokeProcedure(WorklightProcedureInvocationData, Int32) parameters

## Return Value

Type: Task(WorklightResponse)

Returns response from the server.

This method is **Obsolete**.

### 2.7.10 Method RegisterChallengeHandler(GatewayChallengeHandler)

## Syntax

```
public void RegisterChallengeHandler(
    GatewayChallengeHandler challengeHandler
)
```

## Description

You can use this method to register a GatewayChallengeHandler in the client. You must use this method when you implement custom challenge handlers.

You must call this method at the beginning of your application after you initialize the WLClient.

## Parameters

Type	Name	Description
<code>Worklight.GatewayChallengeHandler</code>	<code>challengeHandler</code>	The custom GatewayChallengeHandler instance.

Table 2-31: Method `RegisterChallengeHandler(GatewayChallengeHandler)` parameters

### 2.7.11 Method `RegisterChallengeHandler(SecurityChallengeHandler)`

#### Syntax

```
void RegisterChallengeHandler(
    SecurityCheckChallengeHandler worklightChallengeHandler
)
```

#### Description

You can use this method to register a Challenge Handler in the client. You must use this method when you implement custom challenge handlers, or when you customize the Remote Disable / Notify Challenge Handler.

#### Parameters

Type	Name	Description
<code>Worklight.SecurityCheckChallengeHandler</code>	<code>worklightChallengeHandler</code>	The custom challenge handler instance

Table 2-32: Method `RegisterChallengeHandler(SecurityChallengeHandler)` parameters

### 2.7.12 Method `RemoveGlobalHeader`

#### Syntax

```
void RemoveGlobalHeader(
    string headerKey
)
```

#### Description

You use this method to remove a global header. Then, the header is no longer sent on each request.



**Parameters**

Type	Name	Description
System.string	headerKey	Name of the header.

Table 2-33: Method `RemoveGlobalHeader(String)` parameters**2.7.13 Method ResourceRequest(Uri, String, Int32)****Syntax**

```
public WorklightResourceRequest ResourceRequest (
    Uri uri,
    String method,
    int timeout
)
```

**Description**

Constructs a new resource request with the specified URL, using the specified HTTP method.

**Parameters**

Type	Name	Description
System.Uri	uri	The resource URL, which can be either relative or absolute.
System.String	method	The HTTP method to use.
System.Int32	timeout	The timeout in milliseconds for this request.

Table 2-34: Method `ResourceRequest(Uri, String, Int32)` parameters**Return Value**

Type: `WorklightResourceRequest`

Returns the `WorklightResourceRequest` object.

**2.7.14 Method ResourceRequest(Uri, String, String)****Syntax**

```
public WorklightResourceRequest ResourceRequest (
    Uri uri,
    String method,
    String scope
)
```

### Description

Constructs a new resource request with the specified URL, using the specified HTTP method.

### Parameters

Type	Name	Description
System.Uri	uri	The resource URL, which can be either relative or absolute.
System.String	method	The HTTP method to use.
System.String	scope	The scope required to access the resource.

Table 2-35: Method ResourceRequest(Uri, String, String) parameters

### Return Value

Type: WorklightResourceRequest

Returns the response from the server.

## 2.7.15 Method SetDeviceDisplayName(String)

### Syntax

```
Task<WorklightDeviceDisplayName> SetDeviceDisplayName (
    string DeviceDisplayName
)
```

### Description

Sets friendly device display name for the client in the server.

### Parameters

Type	Name	Description
System.String	DeviceDisplay Name	The friendly device display name.

Table 2-36: Method `SetDeviceDisplayname(String)` parameters

### Return Value

Type: `WorklightDeviceDisplayName`

returns `WorklightDeviceDisplayName` object contains Success or Failure Response.

## 2.8 WorklightProcedureInvocationData Class

### Syntax

```
public class WorklightProcedureInvocationData
```

### Description

This class contains all necessary data to call a procedure.

### Inheritance Hierarchy

System.Object

Worklight.WorklightProcedureInvocationData

### 2.8.1 Constructor WorklightProcedureInvocationData

#### Syntax

```
public WorklightProcedureInvocationData (
    String adapterName,
    String procedureName,
    Object[] parameters
)
```

#### Description

Create a new object to send along with the procedure invocation request.

#### Parameters

Type	Name	Description
System.String	adaptersName	The name of the adapter to invoke
System.String	procedureName	The name of the procedure un the adapter to invoke
System.Object[]	parameters	parameters An array of objects of primitive types ( String, Integer, Float, Boolean, Double). The order of the objects in the array is the order in which they are sent to the adapter.

Table 2-37: Constructor WorklightProcedureInvocationData(String, String, Object[])  
parameters

## 2.8.2 Property AdapterName

### Syntax

```
public string AdapterName { get; }
```

### Description

Name of the adapter being invoked

### Property Value

Type: String

## 2.8.3 Property CompressResponse

### Syntax

```
public bool CompressResponse { get; }
```

### Description

Enable or disable compression of the response data

### Property Value

Type: Boolean

## 2.8.4 Property Parameters

### Syntax

```
public Object[] Parameters { get; }
```

### Description

Procedure invocation parameters

### Property Value

Type: Object[]

## 2.8.5 Property ProcedureName

### Syntax

```
public string ProcedureName { get; }
```

### Description

Name of the procedure in the adapter being invoked

### Property Value

Type: String

## 2.8.6 Method getInvocationDataMap

### Syntax

```
public Dictionary<string, string> getInvocationDataMap()
```

### Description

Get the Invocation data.

### Return Value

Type: Dictionary<String, String>

Returns all parameters as a map.

## 2.9 WorklightRequestOptions Class

### Description

This class contains two objects which are used as part of procedure invocation. The two objects are:

- Timeout-int: the time, in milliseconds, for this invokeprocedure to wait before it fails. The default timeout is 10,000 milliseconds. To disable the timeout, set this parameter to 0.
- InvocationContext-Object: an object that is returned with WLResponse. You can use this object to distinguish different invokeprocedure calls.

These objects are set using the WorklightRequestOptions property.

### Syntax

```
public class WorklightRequestOptions
```

### Inheritance Hierarchy

System.Object

Worklight.WorklightRequestOptions

## 2.9.1 Constructor WorklightRequestOptions

### Description

Initializes a new instance of the WorklightRequestOptions class

### Syntax

```
public WorklightRequestOptions()
```

## 2.9.2 Property InvocationContext

### Description

An object that is returned with response. You can use this object to distinguish different InvokeProcedure calls.

### Syntax

```
public Object InvocationContext { get; set; }
```

### Property Value

Type:Object

### 2.9.3 Property Timeout

The time, in milliseconds, for InvokeProcedure to wait before it fails. The default timeout is 10,000 milliseconds. To disable the timeout set this parameter to 0.

#### Syntax

```
public int Timeout { get; set; }
```

#### Property Value

Type: Int32

## 2.10 WorklightResourceRequest Class

#### Syntax

```
public abstract class WorklightResourceRequest
```

#### Description

This class encapsulates a resource request. The resource may be an adapter on the MobileFirst Server, or an external resource. The class provides several 'send' methods, with different inputs for the body of a request. In addition, the 'send' methods support two types of response listener:

- **WLResponseListener:** The onSuccess method of this listener is called and provided with an instance of the WLResponse class. The content of the response is be read into the WLResponse instance by the platform, and will be accessible through methods of WLResponse. In case of a failure, the onFailure method of the listener is called and provided with an instance of the WLFailResponse class, that will contain all the information about the failure.
- **WLHttpResponseListener:** The onSuccess method of this listener is called and provided with the original HTTP response object that was received from the server. The platform does not attempt to read or parse the response in any way. In case of a failure the onFailure method is called and provided with either the response from the server if one was received, or the exception that was thrown during the execution of this request.

Regardless of what type of listener was used, a successful response is any response with a status in the 2xx range. These responses are delivered to the onSuccess method. A response with a 4xx or 5xx status is considered a failure, and is delivered to the onFailure method.

#### Inheritance Hierarchy

System.Object

Worklight.WorklightResourceRequest

### 2.10.1 Property AllHeaders

#### Syntax

```
public abstract WebHeaderCollection AllHeaders { get; set; }
```

#### Description

Returns all the headers that were set for this resource request.

#### Property Value

Type: WebHeaderCollection

A WebHeaderCollection of headers.

### 2.10.2 Property HeaderNames

#### Syntax

```
public abstract string[] HeaderNames { get; }
```

#### Description

Returns the names of all the headers that were set for this resource request.

#### Property Value

Type: String[]

A string array containing the header names.

### 2.10.3 Property QueryParameters

#### Syntax

```
public abstract IDictionary<string, string> QueryParameters { get; set; }
```

#### Description

Gets or sets the query parameters set for this resource request.

#### Property Value



Type: IDictionary(String,String)

A Dictionary containing the query parameters.

#### **2.10.4 Property Timeout**

##### **Syntax**

```
public abstract int Timeout { get; set; }
```

##### **Description**

Gets or sets the timeout in milliseconds for this resource request.

##### **Property Value**

Type: Int32

The timeout for this resource request.

#### **2.10.5 Property URL**

##### **Syntax**

```
public abstract Uri URL { get; }
```

##### **Description**

Returns the URL for this resource request.

##### **Property Value**

Type: Uri

The Uri object representing the path for this resource request.

#### **2.10.6 Method AddHeader**

##### **Syntax**

```
public abstract void AddHeader(  
    WebHeaderCollection header  
)
```

##### **Description**

Adds a header to this resource request. This method allows response headers to have multiple values.

**Parameters**

Type	Name	Description
System.Net.WebHeaderCollection	Header	The header name to be added on request.

Table 2-38: Method `AddHeader(String, String)` parameters**2.10.7 Method GetFirstHeader****Syntax**

```
public abstract KeyValuePair<string, string> GetFirstHeader(
    String headerName
)
```

**Description**

Returns the first header for this resource request with the given name.

**Parameters**

Type	Name	Description
System.Net.WebHeaderCollection	headerName	The header name of the header data to be retrieved.

Table 2-39: Method `GetFirstHeader(WebHeaderCollection)` parameters**Field Value**

Type: `KeyValuePair(String, String)`

The first header corresponding the given name, or null if no such headers exist.

**Return Value**

Type: `KeyValuePair(String, String)`

Returns the response which contains header name and value.

**2.10.8 Method GetHeaders****Syntax**

```
public abstract WebHeaderCollection GetHeaders (  
    String headerName  
)
```

### Description

Returns all the headers for this resource request that have the given name.

### Parameters

Type	Name	Description
System.String	headerName	The name of the headers to return

Table 2-40: Method `GetHeaders(String)` parameters

### Return Value

Type: `WebHeaderCollection`

An array of Headers.

## 2.10.9 Method RemoveHeaders

### Syntax

```
public abstract void RemoveHeaders (  
    String headerName  
)
```

### Description

Removes all the headers for this resource request with the given name.

### Parameters

Type	Name	Description
System.String	headerName	The name of the headers to remove

Table 2-41: Method `RemoveHeaders(String)` parameters

## 2.10.10 Method Send

### Syntax

```
public abstract Task<WorklightResponse> Send()
```

### Description

Send this resource request asynchronously, without a request body.

### Return Value

Type: Task(WorklightResponse)

A WorklightResponse object with the response from the server

## 2.10.11 Method Send(JObject)

### Syntax

```
public abstract Task<WorklightResponse> Send(  
    JObject json  
)
```

### Description

Send this resource request asynchronously, with the given JSON object as the request body. If no content type header was set, this method will set it to "application/json".

### Parameters

Type	Name	Description
JObject	json	The JSON object to put in the request body.

Table 2-42: Method Send(JObject) parameters

### Return Value

Type: Task(WorklightResponse)

A WorklightResponse object with the response from the server

## 2.10.12 Method Send(Byte[])

### Syntax

```
public abstract Task<WorklightResponse> Send(  
    byte[] data  
)
```

## Description

Send this resource request asynchronously, with the content of the given byte array as the request body. Note that this method does not set any content type header, if such a header is required it must be set before calling this method.

## Parameters

Type	Name	Description
System.Byte[]	Data	The byte array containing the request body.

Table 2-43: Method `Send(Byte[])` parameters

## Return Value

Type: Task(WorklightResponse)

A WorklightResponse object with the response from the server

### 2.10.13 Method `Send(Dictionary(String, String))`

#### Syntax

```
public abstract Task<WorklightResponse> Send(
    Dictionary<string, string> formParameters
)
```

## Description

Send this resource request asynchronously, with the given form parameters as the request body. If no content type header was set, this method will set it to "application/x-www-form-urlencoded".

## Parameters

Type	Name	Description
System.Collections.Generic.Dictionary<String, String>	formParameters	The parameters to put in the request body.

Table 2-44: Method `Send(Dictionary(String,String))` parameters

## Return Value

Type: Task(WorklightResponse)

A WorklightResponse object with the response from the server

### 2.10.14 Method Send(String)

#### Syntax

```
public abstract Task<WorklightResponse> Send(  
    string requestBody  
)
```

#### Description

Send this resource request asynchronously, with the given string as the request body. If no content type header was set, this method will set it to "application/json".

#### Parameters

Type	Name	Description
System.String	requestBody	The string to put in the request body.

Table 2-45: Method Send(String) parameters

## Return Value

Type: Task(WorklightResponse)

A WorklightResponse object with the response from the server

### 2.10.15 Method SetHeader

#### Syntax

```
public abstract Task<WorklightResponse> Send(  
    string requestBody  
)
```

#### Description

Sets a header for this resource request. If the header had already been set, the new value overwrites the previous one.

**Parameters**

Type	Name	Description
System.Collections.Generic.KeyValuePair(String, String)	header	The header to set.

Table 2-46: Method `SetHeader(String, String)` parameters**2.10.16 Method SetHeaders****Syntax**

```
public abstract void SetHeaders(
    WebHeaderCollection headers
)
```

**Description**

Sets the headers.

**Parameters**

Type	Name	Description
System.Net.WebHeaderCollection	headers	Headers.

Table 2-47: Method `SetHeaders(WebHeaderCollection)` parameters**2.10.17 Method SetQueryParam****Syntax**

```
public abstract void SetQueryParam(
    String name,
    String value
)
```

**Description**

Sets the value of the given query parameter name to the given value. If no such parameter exists, it will be added.

## Parameters

Type	Name	Description
System.String	Name	The name of the parameter to set.
System.String	Value	The value of the parameter to set

Table 2-48: Method `SetQueryParameter(String, String)` parameters

## 2.11 WorklightResponse Class

### Syntax

```
public class WorklightResponse
```

### Description

This class contains the result of a procedure invocation.

### Inheritance Hierarchy

System.Object

Worklight.WorklightResponse

### 2.11.1 Property HTTPStatus

#### Syntax

```
public int HTTPStatus { get; }
```

#### Description

This method retrieves the HTTP status from the response.

#### Property Value

Type: Int32

### 2.11.2 Property Message

#### Syntax

```
public string Message { get; }
```



## Description

This method returns an error message that is for the developer, and not necessarily suitable for the user. Particularly useful in case of failure. In case of Success you may not get any message.

## Property Value

Type: String

### 2.11.3 Property ResponseJSON

#### Syntax

```
public JObject ResponseJSON { get; }
```

#### Description

The JSON object obtained from response text.

#### Property Value

Type: JObject

### 2.11.4 Property ResponseText

#### Syntax

```
public string ResponseText { get; }
```

#### Description

This method retrieves the original response text from the server.

#### Property Value

Type: String

### 2.11.5 Property Success

#### Syntax

```
public bool Success { get; }
```

#### Description

Is the response from the server a success or a failure.

## Property Value

Type: Boolean

## Appendix A. Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing*  
*IBM Corporation*  
*North Castle Drive, MD-NC119*  
*Armonk, NY 10504-1785*  
*US*

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

*Intellectual Property Licensing*  
*Legal and Intellectual Property Law*  
*IBM Japan Ltd.*  
*19-21, Nihonbashi-Hakozakicho, Chuo-ku*  
*Tokyo 103-8510, Japan*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new

editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*IBM Director of Licensing*  
*IBM Corporation*  
*North Castle Drive, MD-NC119*  
*Armonk, NY 10504-1785*  
*US*

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).

Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. \_enter the year or years\_.

## Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company products or service names may be trademarks or service marks of others.

This document may not be reproduced in whole or in part without the prior written permission of IBM.

## Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

## Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

## Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

## Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

## **IBM Online Privacy Statement**

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect session information (generated by the application server). These cookies contain

no personally identifiable information and are required for session management. Additionally, persistent cookies may be randomly generated to recognize and manage anonymous users. These cookies also contain no personally identifiable information and are required.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent. For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at [www.ibm.com/privacy](http://www.ibm.com/privacy) and IBM's Online Privacy Statement at [www.ibm.com/privacy/details](http://www.ibm.com/privacy/details) the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at [www.ibm.com/software/info/product-privacy](http://www.ibm.com/software/info/product-privacy).

## Appendix B. Support and comments

For the entire IBM MobileFirst™ Platform documentation set, training material and online forums where you can post questions, see the IBM® website at:

[www.ibm.com/mobile-docs](http://www.ibm.com/mobile-docs)

### Support

Software Subscription and Support (also referred to as Software Maintenance) is included with licenses purchased through Passport Advantage® and Passport Advantage Express. For additional information about the International Passport Advantage Agreement and the IBM International Passport Advantage Express Agreement, visit the [Passport Advantage website \(www.ibm.com/software/passportadvantage\)](http://www.ibm.com/software/passportadvantage).

If you have a Software Subscription and Support in effect, IBM provides you assistance for your routine, short duration installation and usage (how-to) questions, and code-related questions. For additional details, consult your [IBM Software Support Handbook \(www.ibm.com/support/handbook\)](http://www.ibm.com/support/handbook).

### Comments

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this document. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact



you about the issues that you state.

Thank you for your support.

If you would like a response from IBM, please provide the following information:

- Name
  - Address
  - Company or Organization
  - Phone No.
  - Email address
-