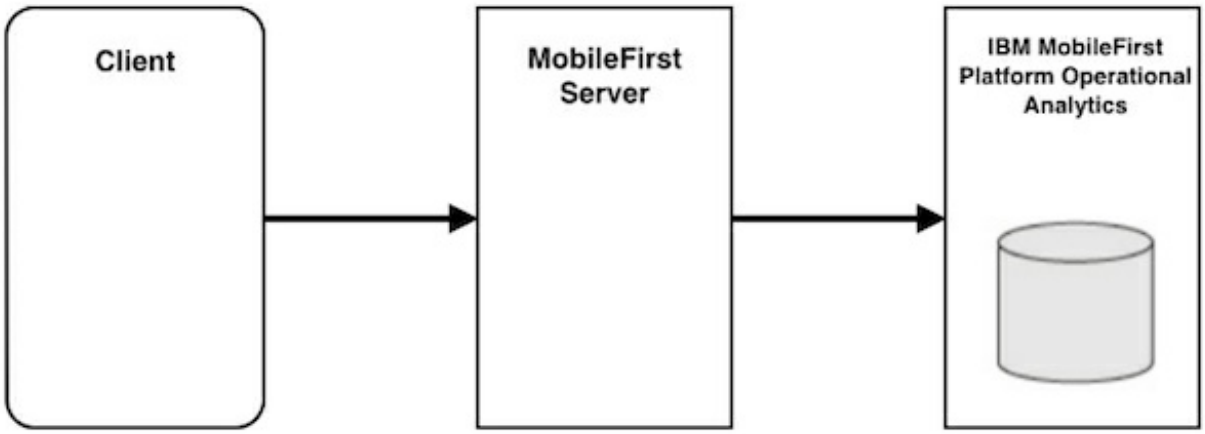
Operational Analytics

Introducing IBM MobileFirst Platform Operational Analytics

MobileFirst Operational Analytics collects data about applications, adapters, devices, and logs to give a high-level view of the client interaction with the IBM MobileFirst Platform Server, and to enable problem detection.



In IBM MobileFirst Platform Foundation V6.3, MobileFirst Operational Analytics is delivered as a WAR file that can be deployed to the following supported application servers:

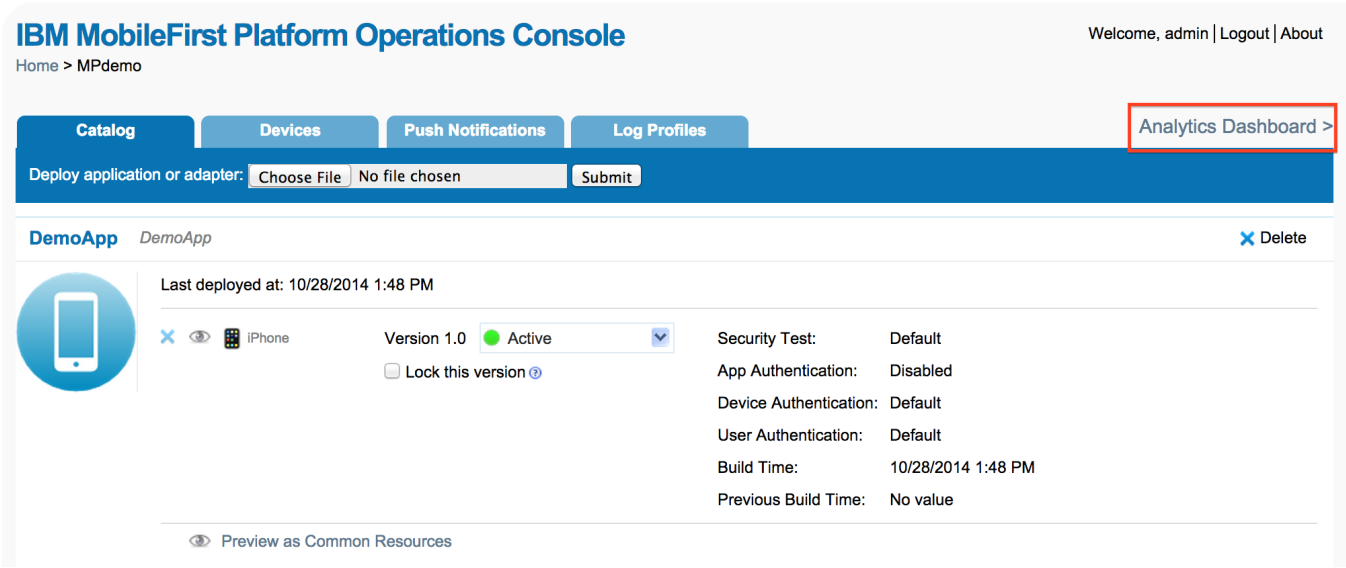
- Liberty
- WebSphere® Application Server
- Tomcat

In MobileFirst Studio, the WAR file is installed and available by default in the embedded Liberty server.

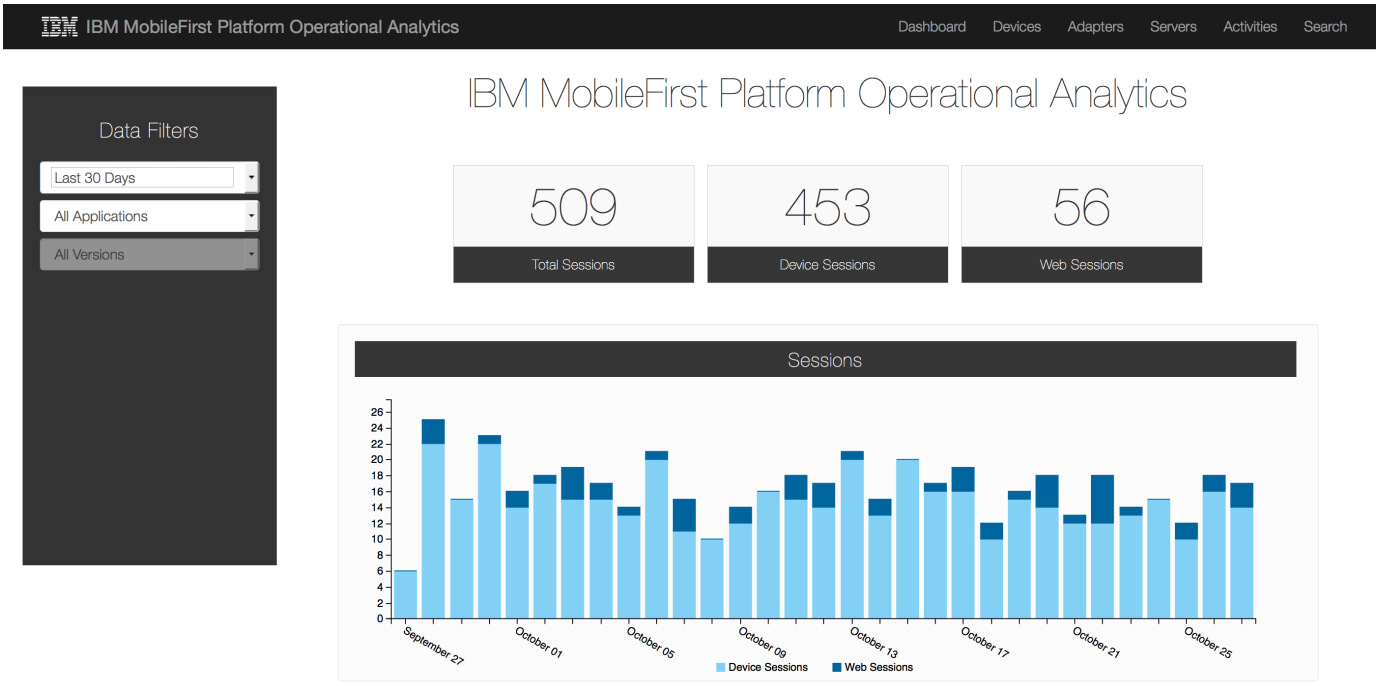
Viewing the Analytics Dashboard - Configurations

The `wl.analytics.url` property must be set to access the analytics dashboard. In MobileFirst Studio, this property is automatically set.

After the property is set, the **Analytics Dashboard** link appears in the MobileFirst Operations Console.



Click the **Analytics Dashboard** link to open up the dashboard in a new window.



Capturing data

Five different types of analytics events are captured by the MobileFirst Operational Analytics. The next sections describe how these events are captured.

Application Activities

- Client initializations with the server
- Adapter calls

Notification Activities

- Push notifications

Server Logs

- Server Events
- Server stack traces

Client Logs

- Debug logs
- Crashes
- Custom Events
- Network latency information

Capturing data - Application Activities

When an application activity occurs, the event is captured automatically and forwarded to the MobileFirst Operational Analytics.

- The following API call results in a session hit that is visualized on the MobileFirst Operational Analytics:

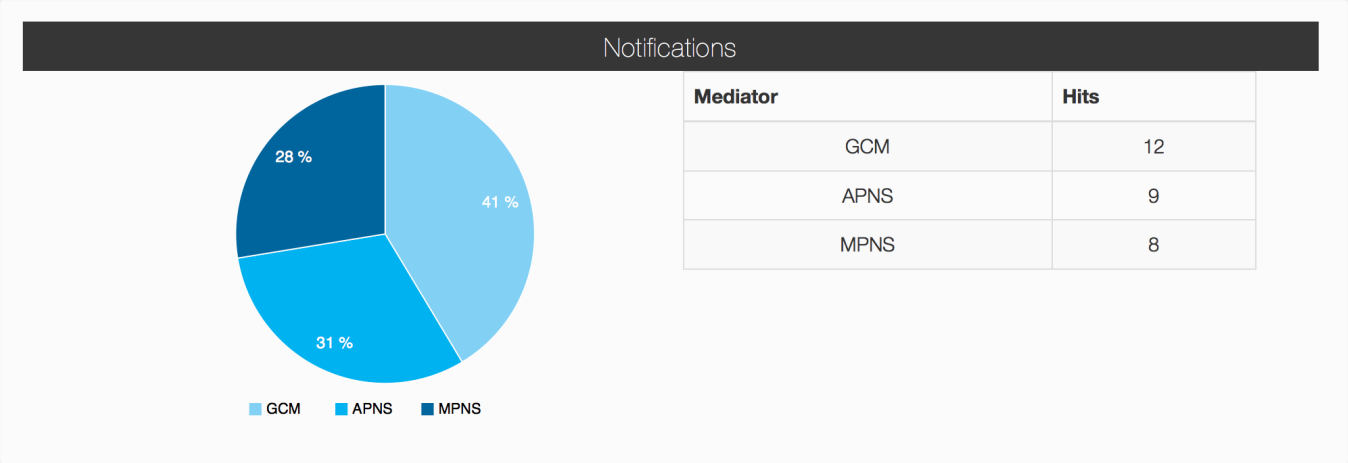
```
// a 'session hit' will be recorded upon a successful connection
WL.Client.connect();
```

- The following API call results in an adapter hit that is visualized on the MobileFirst Operational Analytics:

```
// an 'adapter hit' will be recorded upon a successful adapter invocation
WL.Client.invokeProcedure({...});
```

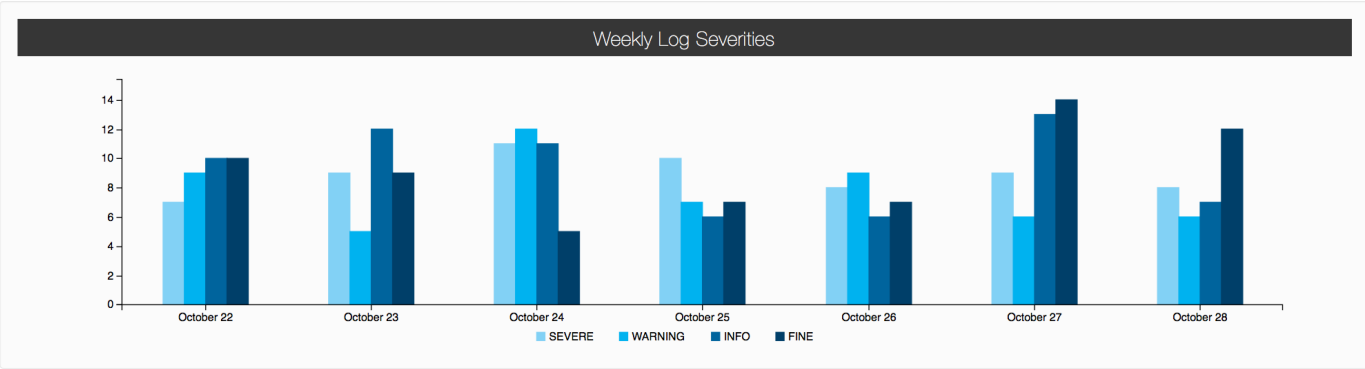
Capturing data - Notification Activities

When a push notification occurs, the event is captured automatically and forwarded to the MobileFirst Operational Analytics.



Capturing data - Server Logs

The log data that is generated by the MobileFirst Server is automatically forwarded to the MobileFirst Operational Analytics where the data can be searched and downloaded.



Server Logs

Date	Severe	Warning	Info	Fine	Download
Tuesday, October 28, 2014	8	6	7	12	Download
Monday, October 27, 2014	9	6	13	14	Download
Sunday, October 26, 2014	8	9	6	7	Download
Saturday, October 25, 2014	10	7	6	7	Download
Friday, October 24, 2014	11	12	11	5	Download
Thursday, October 23, 2014	9	5	12	9	Download
Wednesday, October 22, 2014	7	9	10	10	Download

To disable this behavior, the `wl.analytics.logs.forward` property must be set to `false`.

Capturing data - Client Logs

A MobileFirst application can be instrumented with client logs to record client debugging information and events.

The following APIs are available to create client logs that are then forwarded to the MobileFirst Operational Analytics where they can be searched and downloaded.

```
// Set the log level to trace so all logs are captured
WL.Logger.config({"level": "TRACE"});
// Create a client side log that is persisted locally until sent to the server
WL.Logger.trace("Create a client log at the TRACE level.");
WL.Logger.debug("Create a client log at the DEBUG level.");
WL.Logger.info("Create a client log at the INFO level.");
WL.Logger.warn("Create a client log at the WARN level.");
WL.Logger.error("Create a client log at the ERROR level.");
WL.Logger.fatal("Create a client log at the FATAL level.");
```

Analytics Logs

Client-side logs are captured based on the logging level that is set on the client. If you want to create analytics logs that are always captured regardless of the logging level, you can use the `WL.Analytics` API.

```
// Create an analytics log message
WL.Analytics.log("Analytics log message");
// Create a custom activity
WL.Analytics.log({_activity: "customActivity"});
```

Sending data

Logs that are captured by the client-side logging APIs and the `WL.Analytics` APIs are sent to the server automatically upon a successful server connection or a successful adapter call.

```
// Logs sent upon successful connection
WL.Client.connect();
// Logs sent upon successful adapter invocation
WL.Client.invokeProcedure({...});
```

This automatic behavior can be disabled by using the following call:

```
// Disable automatic sending of client and analytics logs
WL.Logger.config({autoSendLogs: false});
```

If you want to send this data more frequently, you can use the following API calls:

```
// Send client debug logs
WL.Logger.send();
// Send analytics logs
WL.Analytics.send();
```