
ORACLE®

Oracle Intelligent Bots Advanced Training

Front-End Channels

ORACLE

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Topic Agenda

1. Introduction to Bots Channels
2. Android, iOS and Web Client SDKs
3. HTTP Webhook
4. Demos

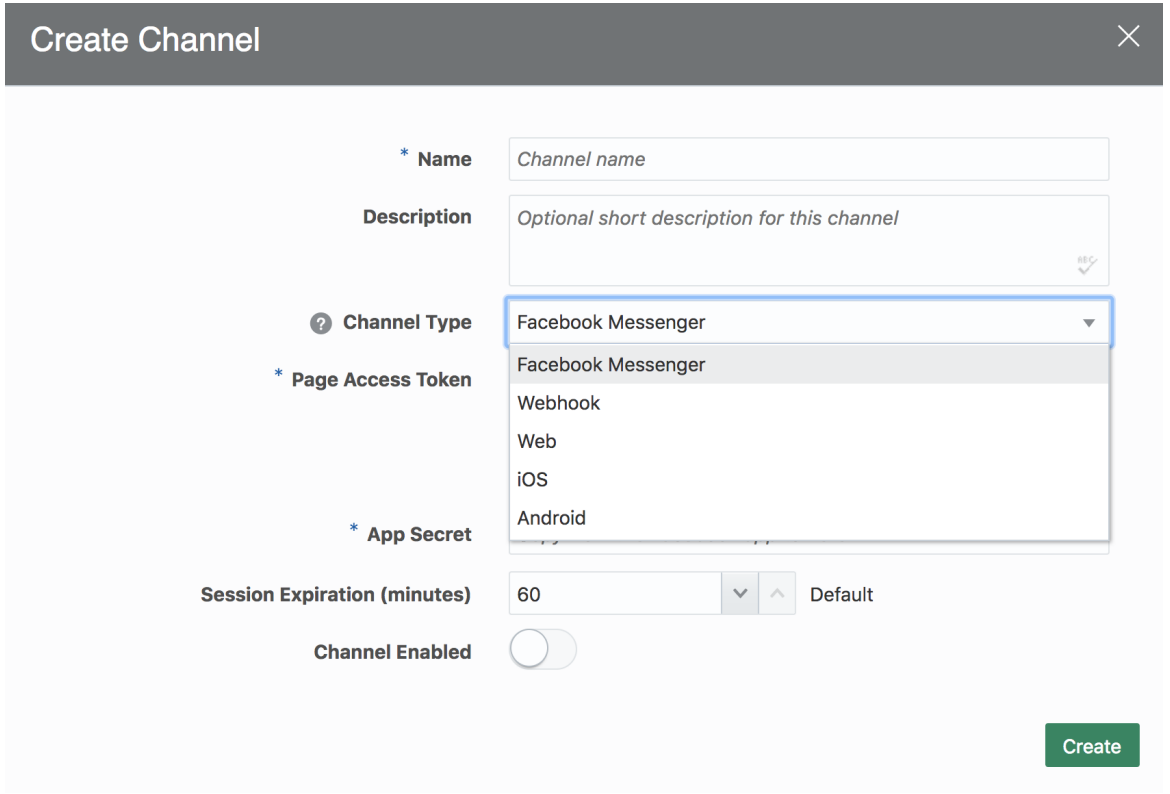
Supporting Users with Multiple Messaging Platforms

- The more messaging services you need to support the more infrastructure you need to put in place
- Every messaging service has a unique set of features
  - Text-only
  - Images
  - Carousel

Channels integrate messaging services with the bot platform

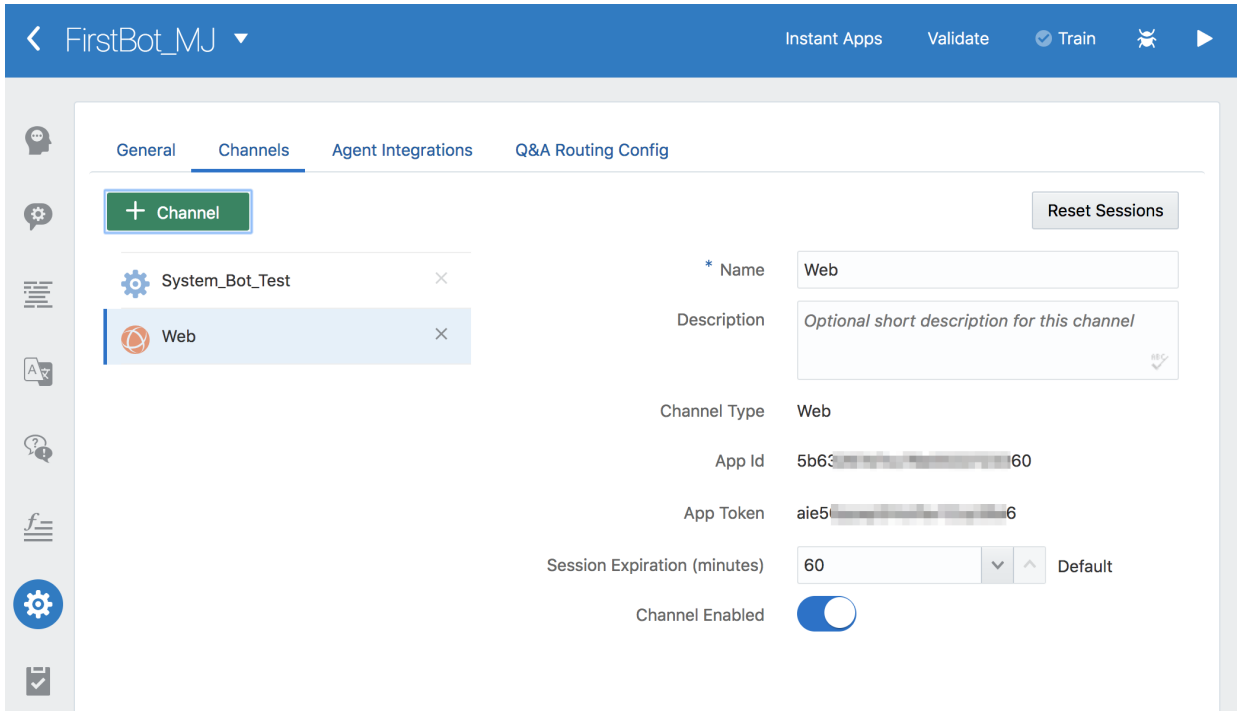
Oracle Intelligent Bots Channels

- Current available channels are:
  - Facebook Messenger
  - Web
  - iOS
  - Android
  - Generic HTTP Webhook



Creating New Channels

- To create a new Channel in your Bot
  - Go to Settings, select the Channels tab
  - Click the Add Channel button
  - Provide required information
  - Select Channel Type
  - Switch on the Channel Enabled toggle

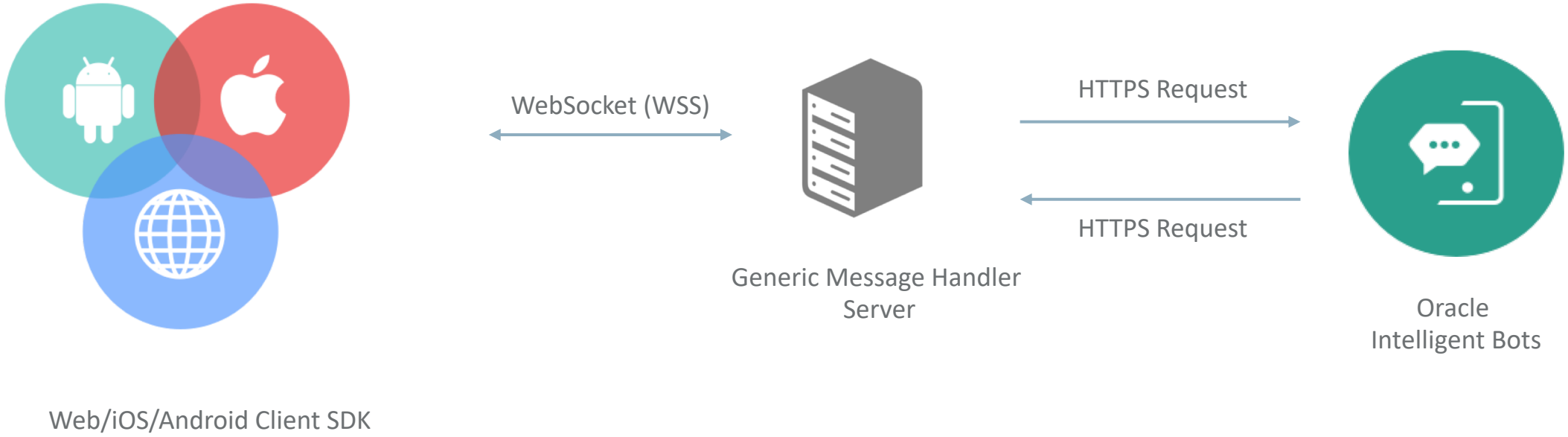


Oracle provides client SDKs to integrate Oracle Intelligent Bots with Android, iOS and Web applications

ORACLE

Image courtesy of Farbentek at FreeDigitalPhotos.net

How the Client SDK Works



Oracle Intelligent Bots - Client SDKs

- Customizable messenger SDKs that can be added to Android, iOS, Web page or hybrid mobile app
- Free download from
  - http://www.oracle.com/technetwork/topics/cloud/downloads/mobile-suite-3636471.html
- Simple steps to add the client SDK to your app

Adding the Web Client SDK to Your Web Site

- Include the Web Messenger on your web page
  - Add the following code towards the end of the <head> section

```
<script>
  !function(e,t,n,r){
  function s(){

    //implementation : see client – sdk documentation

  }(window,document,"Bots", "<sdk-folder-url>")

</script>
```

Replace <sdk-folder-url> with the URL where the SDK is hosted

Adding the Web Client SDK to Your Web Site

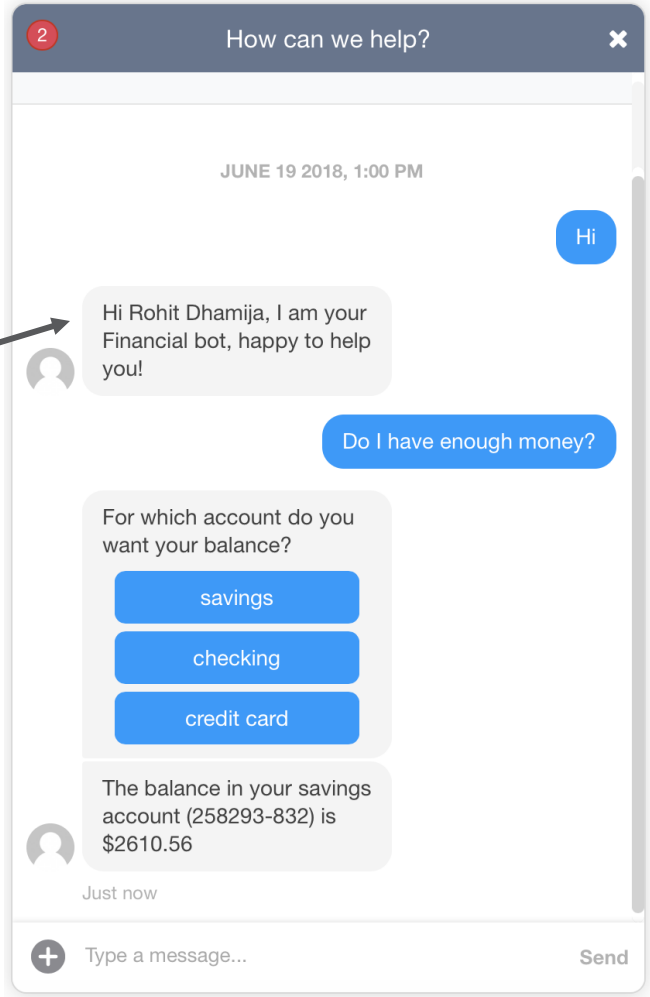
- Initialize the Web Messenger where you want to enable chat capabilities

```
<script>
   Bots.init({
      appId: "your_app_id"


    }).then(function (res){
  });
</script>
```

Adding the Web Client SDK to Your Web Site

- Update User Profile Information

```
<script>
   Bots.init({
      appId: "your_app_id"

   }).then(function (res){
      Bots.updateUser(
          {
              "givenName":"Rohit",
              "surname":"Dhamija",
               "properties": {
                 "userCustomVariable1":"Lord",
                 "userCustomVariable2":"Commander"
               }
          }
      ).catch(function (err) {
          console.error(err);
      });
   });
</script>
```



Adding the Web Client SDK to Your Web Site

- Accessing User Profile Information in the Dialog Flow

```
states:
  outputGenericProfileVariables:
    component: "System.Output"
    properties:
      keepTurn: true
      text: "Generic profile variables \n\n
             First name: ${profile.firstName.value} \n
             Last name: ${profile.lastName.value}"
    transitions: {}

  outputCustomProfileVariables:
    component: "System.Output"
    properties:
      keepTurn: true
      text: "Custom variables \n\n
             userCustomVariable1: ${profile.userCustomVariable1.value} \n
             userCustomVariable2: ${profile.userCustomVariable2.value}"
    transitions:
      return: "done"
```

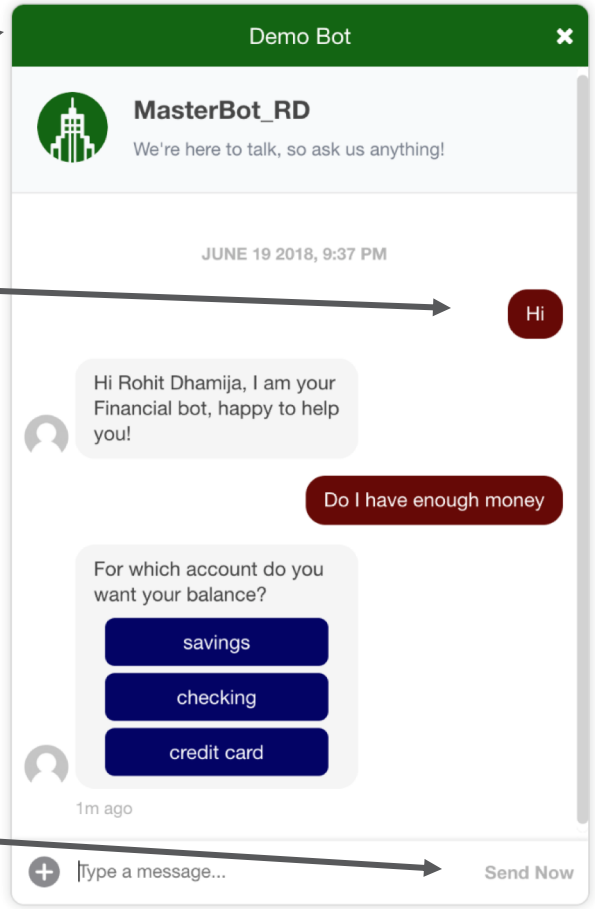
Customizing the Web Messenger SDK

- Icons
- Sounds
- Colors

- Locale
- Text
- Style
- Date Localization

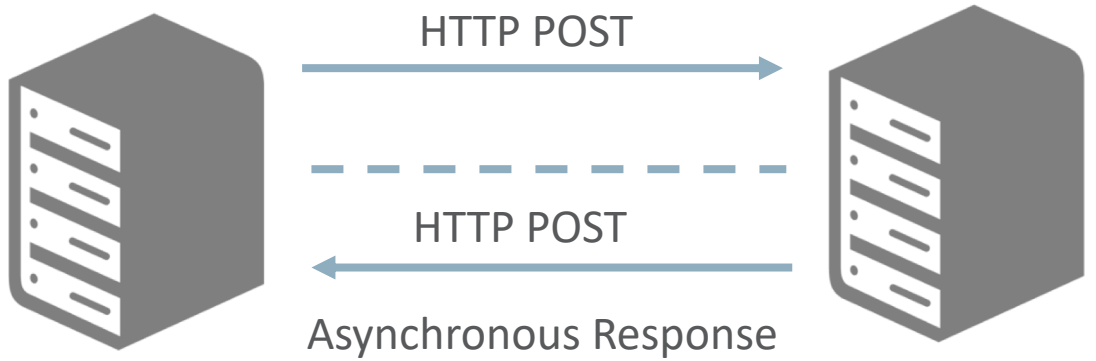
Customize Web Messenger Demo

```
<script>
    Bots.init({
      appId: "5b2253aa1f0cd100221d3974",
      customColors: {
      brandColor: '006600',
      conversationColor: '660000',
      actionColor: '000066',
      },

     customText: {
      headerText: 'Demo Bot',
      inputPlaceholder: 'Type a message...',
      introductionText: 'We\'re here to talk, so ask us anything!',
      messageSending: 'Sending...',
      messageDelivered: 'Delivered!',
      sendButtonText: 'Send Now',
     }
   }).then(function (res){
 }); </script>
```

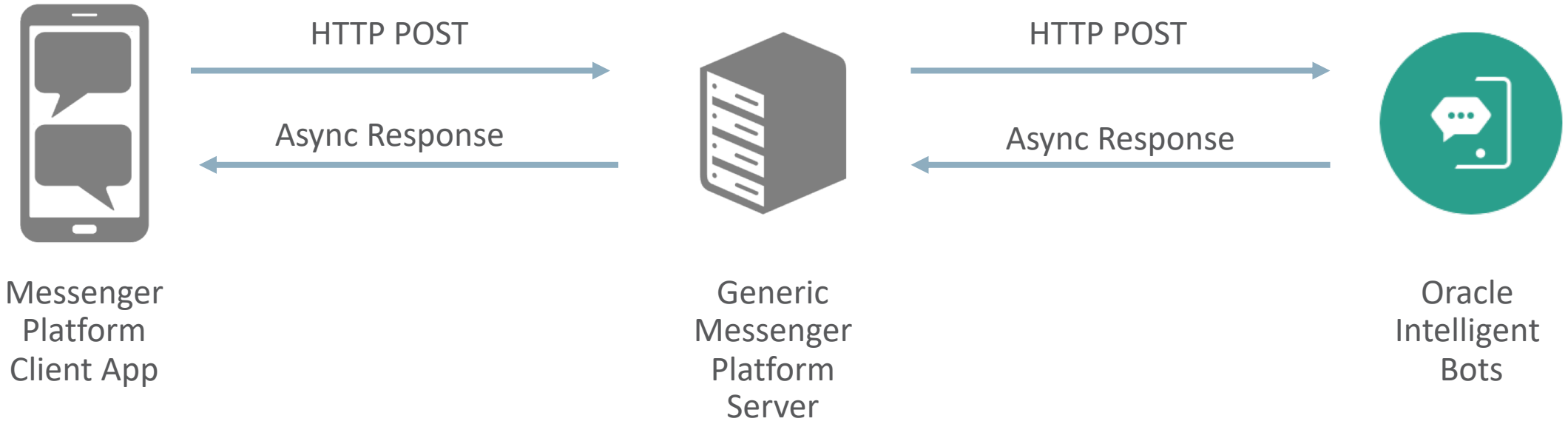


Generic HTTP Webhooks

- Webhooks are user-defined callbacks made with HTTP POST
- Used for server to server integration
- Triggered by application events
- Pub/sub like, commonly utilizes REST APIs
- Servers don't poll each other
  - Server asynchronously sends event updates via HTTP POST



Generic HTTP Webhook Channel



Generic HTTP Webhook Support

- Generic channel mechanism
  - Supporting other messenger platforms
    - SMS, Alexa, Slack, etc
    - Virtually any messaging platform that supports HTTP Webhooks
- Through the WebHook channel
  - Bot channel publishes an HTTP Endpoint to receive messages
  - You define a response HTTP Endpoint
  - Bot will send response messages back to your server

- To verify messages the Intelligent Bot publishes a secret key in the WebHook channel
- Caller must supply
  - X-Hub-Signature HTTP header
  - Set to SHA256 signature of payload
  - Secret key used as SHA key
- Uses same mechanism on response
  - Optional for caller to verify payload

Integrated Cloud

Applications & Platform Services


ORACLE®