On This Page

Bundle-based configuration

Some of the configuration for your Instant Game can be made either via the developer portal or via a special JSON file embedded with your game's bundle. When a value is updated in the developer portal, this is a global change for all builds of the game, whereas a bundle-based configuration only changes that value for one specific bundle. This is particularly useful for soft-launching scenarios where two or more bundles can be live at the same time.

This document will explain in detail how to make configuration changes to a specific build.

1. Setup
2. Supported Configurations
3. Reference

Setup

To enable a bundle-specific configuration, include a file called `fbapp-config.json` with your zip file.

```json
{
  "instant_games": {
    "property_1": "VALUE A",
    "property_2": "VALUE B",
    //...
  }
}
```

The file needs to be included in the root of the archive and not in any subfolders, similarly to `index.html`. If this file is found in the bundle, any configuration values it contains will override the values for the global settings in the app dashboard.

Supported Configurations

Platform version

The `platform_version` property allows to set a specific bundle to use our Classic platform (SDK 2.1 and below) or our Rich Gameplay Features, available in SDK 3.0 and above. It accepts the string values `'RICH_GAMEPLAY'` or `'CLASSIC'`

Below is an example of an `fbapp-config.json` file that sets this value.

```
{
  "instant_games": {
    "platform_version": "RICH_GAMEPLAY",
  }
}
```

Custom update templates

This configuration allows you to specify identifiers for each Custom Update sent by your application. You can send custom updates with identifiers from SDK 4.0 and up. This is how the configuration should be setup on `fbapp-config.json`

```
{
  "instant_games": {
    "custom_update_templates": {
      "pass_score": {
        "example": "Kun just scored 100 and passed Alissa's highscore!"
      },
      "play_turn": {
        "example": "Kun just played HELLO. Now it's Alissa's turn!"
      }
    }
  }
}
```

Once these templates are setup in the configuration file, they can be referenced via the `FBInstant.updateAsync()` call via the `template` parameter like in the example below:

```
// game.js
FBInstant.updateAsync({
  action: 'CUSTOM',
  template: 'play_turn',
  text: 'Kun just played HELLO. Now it\'s Alissa\'s turn!',
  image: '...',
  data: '...',
})
```

> Future implementations of this configuration will allow you to specify formatted strings and localization to each of the templates.

Player Stats

Some player statistics stored with the player stats API can be surfaced on Facebook. These statistics can be configured in `fbapp-config.json` as follows:

```
{
  "instant_games": {
    "surfaceable_stats": {
      "rank": {
        "priority": 1,
        "order": 1,
        "label": {
          "localizations": {
            "en_US": "Rank",
            "th_TH": "\u{0E22}\u{0E28}",
          },
          "fallback": "Rank"
        },
        "values": {
          "0": "Unranked",
          "1": "Bronze",
          "2": "Silver",
          "3": "Gold",
        }
      }
    }
  }
}
```

Reference

For the full reference documentation that shows the entire range of options for bundle configuration, click below:

Bundle Configuration Reference

Business Tools

Gaming

Open Source

Publishing

Social Integrations

Virtual Reality

Programs

Developer Circles

F8

FbStart

ThreatExchange

Support

Developer Support

Bugs

Platform Status

Facebook Developers Community Group

News

Blog

Success Stories

Videos

Facebook for Developers Page