

# App Shortcuts - Android

If your app targets Android 7.1 (API level 25) or higher, you can define *shortcuts* to specific actions in your app. These shortcuts can be displayed in a supported launcher. Shortcuts let your users quickly start common or recommended tasks within your app.

Each shortcut references one or more [intents](#), each of which launches a specific action in your app when users select the shortcut. Examples of actions you can express as shortcuts include the following:

- Navigating users to a particular location in a mapping app.
- Sending messages to a friend in a communication app.
- Playing the next episode of a TV show in a media app.
- Loading the last save point in a gaming app.

You can publish the following types of shortcuts for your app:

- *Static shortcuts* are defined in a resource file that is packaged into an APK. Therefore, you must wait until you update your entire app to change the details of these static shortcuts.
- *Dynamic shortcuts* are published at runtime using the [ShortcutManager](#) API. During runtime, your app can publish, update, and remove its dynamic shortcuts.
- *Pinned shortcuts* (supported on API 26) are published at runtime and also use the [ShortcutManager](#) API. During runtime, your app can attempt to pin the shortcut, at which time the user receives a confirmation dialog asking their permission to pin the shortcut. The pinned shortcut appears in supported launchers only if the user accepts the pinning request.
- **Note:** Users can also create pinned shortcuts themselves by copying your app's static and dynamic shortcuts onto the launcher.

**Figure 1.** Using app shortcuts, you can surface key actions and take users deep into your app instantly

You can publish up to five shortcuts (static shortcuts and dynamic shortcuts combined) at a time for your app. Some launcher apps, however, don't show every static and dynamic shortcut you've created for your app.

There is no limit to the number of pinned shortcuts to your app that users can create. Even though your app cannot remove pinned shortcuts, it can still disable them.

## Creating Static shortcuts

To create static shortcuts follow these steps:

1. Create xml in your res/xml folder with the root element `<shortcuts>` and for `<shortcut>` you need to specify `android:shortcutId` - unique Mandatory shortcut ID, This must be a string literal. A resource string, such as `@string/foo`, cannot be used, `android:icon` - Shortcut icon, `android:shortcutShortLabel` - Mandatory shortcut short label - This must be a resource string (it will be displayed if the user add your shortcut to the launcher), `android:shortcutLongLabel` - this will appear next to the icon in the shortcuts menu - This must be a resource string, `intent` - Intent to launch when the user selects the shortcut. `android:action` is mandatory. You can provide multiple intents for a single shortcut so that the last defined activity is launched with the other activities in the [back stack](#). **Note: String resources may not be used within an `<intent>` element.** And the last attribute is `categories` Specify shortcut categories. Currently you must use the string `"android.shortcut.conversation"`. **Note: Please refer to Appendix A for complete xml file example**
2. Add the following `<meta-data>` to the activity with the MAIN action (the launch activity of the app : `<meta-data android:name="android.app.shortcuts" android:resource="@xml/[shortcut file name - from section 1]" />`

## Creating dynamic shortcuts

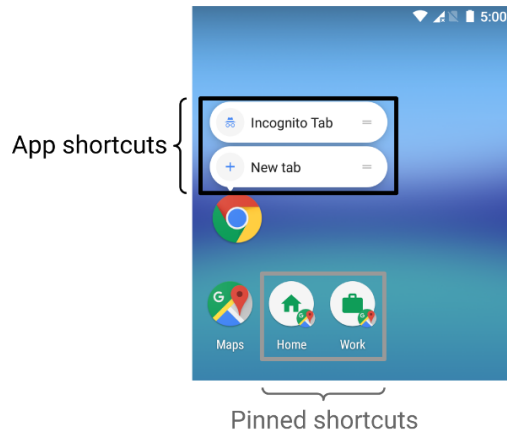
To create app shortcuts dynamically at runtime do the following steps:

1. Get an instance of the **ShortcutManager** using `getSystemService(ShortcutManager.class)`, the shortcut manager allows you to **Publish shortcuts**: Use `setDynamicShortcuts()` to redefine the entire list of dynamic shortcuts, or use `addDynamicShortcuts()` to augment an existing list of dynamic shortcuts, **Update shortcuts**: Use the `updateShortcuts()` method, **Remove shortcuts** : Remove a set of dynamic shortcuts using `removeDynamicShortcuts()`, or remove all dynamic shortcuts using `removeAllDynamicShortcuts()`.
2. Create a **ShortcutInfo** object using the **ShortcutInfo.Builder** passing it the context and the shortcut id, then configure your shortcut like in the xml. Use the `setShortLabel()`, `setLongLabel()`, `setIcon()` and `setIntent()` and use `build()` when finished.

3. Create a `List<ShortcutInfo>` from your shortcut and use the `ShortcutManager.setDynamicShortcuts()` mentioned above.

Please refer to appendix B for dynamic shortcut creation example

## Using Pinned Shortcuts



On Android 8.0 (API level 26) and higher, you can create *pinned shortcuts*. Unlike static and dynamic shortcuts, pinned shortcuts appear in supported launchers as separate icons. **Note:** When you attempt to pin a shortcut onto a supported launcher, the user receives a confirmation dialog asking their permission to pin the shortcut. If the user doesn't allow the shortcut to be pinned, the launcher cancels the request.

To pin a shortcut to a supported launcher using your app, complete the following sequence of steps:

1. Use [`isRequestPinShortcutSupported\(\)`](#) to verify that the device's default launcher supports in-app pinning of shortcuts.
2. Create a [`ShortcutInfo`](#) object in one of two ways, depending on whether the shortcut already exists:
  1. If the shortcut already exists, create a [`ShortcutInfo`](#) object that contains only the existing shortcut's ID. The system finds and pins all other information related to the shortcut automatically.
  2. If you're pinning a new shortcut, create a [`ShortcutInfo`](#) object that contains an ID, an intent, and a short label for the new shortcut.
3. Attempt to pin the shortcut to the device's launcher by calling [`requestPinShortcut\(\)`](#). During this process, you can pass in a [`PendingIntent`](#) object, which notifies your app only when the shortcut is pinned successfully.
4. **Note:** If the user doesn't allow the shortcut to be pinned to the launcher, your app doesn't receive a callback.
5. After a shortcut is pinned, your app can update its contents using the [`updateShortcuts\(\)`](#) method.

Please refer to appendix C or pinned shortcuts example

## Appendix A - shortcuts xml example file

```
<shortcuts xmlns:android="http://schemas.android.com/apk/res/android">

  <shortcut
    android:shortcutId="site"
    android:icon="@drawable/ic_explore_black_24dp"
    android:shortcutShortLabel="@string/compose_shortcut_short_label1"
    android:shortcutLongLabel="@string/compose_shortcut_long_label1">
    <intent
      android:action="android.intent.action.VIEW"
      android:data="http://www.syntax.org.il"/>
    <!-- If your shortcut is associated with multiple intents, include them
         here. The last intent in the list determines what the user sees when
         they launch this shortcut. -->
    <categories android:name="android.shortcut.conversation" />
  </shortcut>

  <shortcut
    android:shortcutId="compose"
    android:icon="@drawable/ic_mail_outline_black_24dp"
    android:shortcutShortLabel="@string/compose_shortcut_short_label2"
    android:shortcutLongLabel="@string/compose_shortcut_long_label2">
    <intent android:action="android.intent.action.VIEW"
      android:data="mailto:eran@syntax.org.il">
      <extra android:name="android.intent.extra.TEXT"
        android:value="What can I help you with?" />
      <extra android:name="android.intent.extra.SUBJECT"
        android:value="Feedback about World Changing App" />
    </intent>
    <!-- If your shortcut is associated with multiple intents, include them
         here. The last intent in the list determines what the user sees when
         they launch this shortcut. -->
    <categories android:name="android.shortcut.conversation" />
  </shortcut>
  <!-- Specify more shortcuts here. -->
</shortcuts>
```

## Appendix B - Dynamic shortcut example

```
ShortcutManager shortcutManager = getSystemService(ShortcutManager.class);
```

```
ShortcutInfo shortcut = new ShortcutInfo.Builder(this, "id1")
    .setShortLabel("Dial")
    .setLongLabel("Dial to last number")
```

```
.setIcon(Icon.createWithResource(this, R.drawable.ic_call_black_24dp))
.setIntent(new Intent(Intent.ACTION_DIAL,
    Uri.parse("tel:" + number)))
.build();
```

```
shortcutManager.setDynamicShortcuts(Arrays.asList(shortcut));
```

## Appendix C - Pinned shortcuts example

```
if (Build.VERSION.SDK_INT >= 26 && shortcutManager.isRequestPinShortcutSupported()) {
    // Assumes there's already a shortcut with the ID "compose".
    // The shortcut must be enabled.
    ShortcutInfo pinShortcutInfo = new ShortcutInfo.Builder(this, "compose").build();

    // Create the PendingIntent object only if your app needs to be notified
    // that the user allowed the shortcut to be pinned. Note that, if the
    // pinning operation fails, your app isn't notified. We assume here that the
    // app has implemented a method called createShortcutResultIntent() that
    // returns a broadcast intent.
    Intent pinnedShortcutCallbackIntent =
        shortcutManager.createShortcutResultIntent(pinShortcutInfo);

    // Configure the intent so that your app's broadcast receiver gets
    // the callback successfully.
    PendingIntent successCallback = PendingIntent.getBroadcast(this, 0,
        pinnedShortcutCallbackIntent, 0);

    shortcutManager.requestPinShortcut(pinShortcutInfo,
        successCallback.getIntentSender());
}
```

## Reference

[App Shortcuts](#) in the Android developer guide.