# Assessing speaking skills pdf

Page 2 Watch 32 Star 338 Fork 198 You can't perform this action at this time. You've signed up with another tab or window. Reboot to update the session. You subscribe to another tab or window. Reboot to update the session. We use additional third-party analytical cookies to understand how you use GitHub.com so we can create the best products. Learn more. We use additional third-party analytical cookies to understand how you use GitHub.com so we can create the best products. You can always update your choices by clicking on the Cookie Preferences at the bottom of the page. For more information, see us that we use important cookies to perform the main functions of a website, such as logging in. Learn More Always Active We use analytics cookies to understand how you use our websites so we can make them better, for example, they are used to gather information about the pages you visit and how much. clicks you need to complete the task. Read more: In the second half of 2021, new apps will have to use Android App Bundle for release in Google Play. New apps larger than 150MB must use Play Feature Delivery or Play Asset Delivery. Android App Bundle is a publishing format that contains all the code and resources for your app, and it sends APK builds and signatures to Google Play. Google Play uses your suite of apps to create and provide optimized APK for each device configuration, so it downloads only the code and resources needed for a particular device to run the app. You no longer have to create, sign, and manage multiple PACs to optimize support for different devices, and users can get smaller and more optimized download packages. For most application projects, creating application packages to support optimized APK is time-consuming. For example, if you've organized the code and app resources as is customary, just use Android Studio or use the command line to create signed Android App Bundles kits and download them to Google Play. Optimized APK can be provided for automatic reuse of benefits. When you publish an app in the app package format, you can also selectively use Play Feature Delivery, which allows you to add feature modules to the app's project. You can eliminate some of the features and resources contained in these modules when users first download and install your app. With the Play Core library, your app can request these modules to be downloaded later. Google Play will only provide code and resources for the device. Game developers who use the app package to publish apps can use Play Asset Delivery: a Google Play solution to distribute a large number of gaming assets, giving developers flexible distribution methods and extremely high performance. Watch the video below to get an overview of why you should use android App Bundle to publish your app. When using the Android App Bundle, the compressed download size limit is currently 150MB. You can't link the app to APK. Apk. If you use Android Studio 3.2 or later, you can build an Android Bundle app in just a few clicks. This page describes the steps to start creating the Android Bundle App, as well as some important concepts related to the app package. Start creating an APP package with APK because you can't deploy it on your device. Instead, it's a publishing format that includes all the code and resources you use in a single build. So once you download a signed package of apps, Google Play has everything you need to build and sign the APK app and make it available to users. If you're using Android Studio, you can build your project as a signed app package in just a few clicks. If you're not using IDE, you can instead create an application package from a command line. Then download the app package to the Play Center to test or publish the app. To create an app package, follow these steps: to create an application package with an asset package, see after testing the app package to create the Android App Bundle, you should check how Google Play uses the Android Bundle app to create APK and how these APKs behave after they are deployed on your device. To test the app package, use any of the following methods: Use the Play Core Library Download If your app contains a feature module, you need to use the Play Core Library Request, Monitoring and Management module to download. To learn more, go downloading modules using the Play Core library. If you want to see the actual application of the library, try the example of the Play Core Library app. Instructions for installing free apps in Android Studio 3.2 or later, you can add a fitting free experience to the app bundle as long as the app is small enough. To learn more about the size limits of the different installations, the free experiences you can create, see the review without installing Google Play. By compressing download size restrictions with Android App Bundle Publishing, you can help users set your app at as little download size as possible and increase the compressed download size limit to 150MB. That is, when a user downloads your app, the total compressed APK (e.g., the basic APK configuration plus the APK configuration) required to install the app should not exceed 150MB. Any subsequent downloads, such as the on-demand loading module (and its APK configuration), must also meet this limit on the compression load size. Asset packages are not limited to this size, but they have other size limits. When you download an app package, you get an error if Center will find that your app or its on-demand features can download more than 150MB. Please note that Android App Bundle does not support APK extensions . . . obb files. So if you encounter this error when you publish an application package, use one of the following resources to reduce the size of the compressed APK download: Be sure to install enableSplit s true for each type of configuration to include the entire APK configuration. This ensures that users only download the code and resources needed to run the app on their devices. Devices. Follow the best practices to further reduce the size of the app. Consider converting features that are only available to some users for modules that your app can download on demand later. Note that this may require a small refactoring of your app, so be sure to try the other suggestions above first. Known problems Following describes the current known problems that arise when creating or delivering an app with Android App Bundle. If you run into a problem that isn't described below, report the error. Some side-loading apps installed (i.e. apps that are not installed in the Google Play Store and do not have one or more apps requiring APK separation) will fail on all Google-certified devices and on Android 10 devices (API Level 29) or later. When you download the app through the Google Play Store, Google ensures that all the necessary components of the app are installed. Using a tool that dynamically changes the resource table, the APK generated from the application package may behave abnormally. That's why when you create an app package, we recommend turning off these tools. The functional modules manifesto should not refer to resources that do not exist in the base module. This is because when Google Play builds a basic APK for the app, the manifesto of all modules merges into the list of major APK. Therefore, if the manifest for the APK base refers to a resource that is not in the APK database, the resource association is violated. Starting with Android Studio 3.2 Canary 14, when you build options for major module changes in your app, the system doesn't automatically select the same build options for functional modules that rely on base modules. So you can get an application build error. Just make sure you choose the same build option for the base module and the other modules that depend on it. Properties that conflict with the properties of a base module or other modules can now be configured in the build configuration of the functional module. For example, you can install buildTypes.release.debuggable to make this true in the base module and set it to be false in the function module. Such conflicts can cause build and launch problems. Note that the default function module inherits some build configurations from the base module. Therefore, it's important to understand which configurations should be saved and lowered in the function module build configuration. To download the feature module, the device must install the latest version of the Play Store app. So if your app includes module features, a small number of user downloads can go back to one optimized multi-APK that same download experience for Android 4.4 devices (API Level 20) and earlier. The Android App Bundle Android App Bundle format is a file you download to Google Play (file extension .aab). An application package is a signed binary that organizes the application code and resources into modules, as shown in Figure 1. The code and resources of each module are organized in the same way as in APK, which makes sense because each module can be created as a separate APK. Google Play then uses a suite of apps to create a variety of AKS, such as the basic APK features that are available to users. Users. APK and multi-APK (multi-APK for devices that do not support the APK division). A catalog identified in blue, such as drawable/, values/, and lib/directory, represents the code and resources that Google Play uses to create an APK configured for each module. Figure 1. Content for Android App Bundle with one base module, two function modules and two asset packs. Note: You need to create an application package for each unique application or applicationID. That is, if you use product options to create multiple versions of the app from the same app project, and each version uses a unique applicationID, you need to build a separate suite of apps for each version of the app. The following list describes some of the files and directories of the app package in more detail: basic/, feature1, and feature2/: Each of the top-level directories represents a different application module. The main modules of the app are always included in the basic app package catalog. However, the name in each function module's catalog is listed by a split property in the module's manifest. For more information, see a list of feature modules. asset_pack_1/asset_pack_2/: For large apps or games that require a lot of graphic processing, you can modalise assets into asset packages. The asset pack is perfect for gaming because of its high size cover. You can set up how and when individual asset packages are loaded onto the device in three distribution modes: installation distribution, on-demand distribution, and on-demand distribution. All asset packages are placed on Google Play and are available on Google Play. To learn more about how to add an asset package to an application package, see BUNDLE-METADATA/: This catalog contains metadata files that contain information useful to a tool or store. Such metadata files may contain a complete list of DeX files that ProGuard cards are still in use. The files in this directory are not packaged in your app's APK. Module Protocol Buffer (.pb) files: These files provide metadata to help explain the contents of each app module for each app store, such as Google Play. For example, BundleConfig.pb provides information about the package itself, such as the version of the build tool used to create the application package, and native.pb and resources.pb illustrates the code and resources in each module, which is useful when Google Play optimizes APK for different device configurations. Manifesto/: Unlike APK, the app package stores AndroidManifest.xml files for each module in this separate catalog. dex/: Unlike APK, the app package stores DEX files for each module in this separate catalog. Res/, lib/and assets/: These catalogs are identical to catalogs in a typical APK. downloading the Google Play app package checks directories and packages of only files that meet the needs of the target device configuration while keeping the file path. Root/: The files stored in this catalog are then repositioned to the root of any APK that contains the module in which the catalog is located. For example, a basic/root/app package directory may contain your app for use. Use. Java-based resources are loaded. These files are then repositioned to the root of each multi-APK generated by your app's basic APK and Google Play. The paths in this catalog have also been preserved. That is, the catalog (and its subdirection) also moves to the root of the APC. Note: If the contents of this directory conflict with other files and directories under the APK root, Play Center rejects the entire package of applications when downloaded. For example, you can't include a root/lib/catalogue because it contradicts the lib catalog that each APK already contains. Split APK Review The main component needed to ensure optimized applications is the split APK mechanism available on Android 5.0 (API level 21) and beyond. The APK division is very similar to the usual APK and contains DEX codes, resources and Android manifests. However, the Android platform can consider several separated APK installed as a single application. That is, you can install several separated PKEs that share code and resources and look like an app installed on your device. The advantage of separating APK is the ability to divide the APK monosodi into smaller standalone packages that can be installed on demand on the user's device. Monozodial APK is an APK that contains code and resources for all features and configurations of devices supported by the app. For example, a divided APK may contain code and resources for additional features that few users need, while a divided APK may only contain resources for a particular language or screen density. These separated APKs can be downloaded and installed separately at the user's request or on the device's needs. Here are the different types of PCs that can be installed together on your device to create a complete experience. The following few sections of this page show how to set up an app to support these PACs. Basic APK: This APK contains code and resources that are available to all other separated AKKs and provides basic application functionality. When a user requests an app to download, THE APK is downloaded and installed first. This is due to the fact that only the inventory of the basic APK contains a full declaration of the application services, content providers, permissions, platform version requirements, and system dependencies. Google Play generates a basic APK for the application based on the project application module, the main module. If you want to reduce the initial download size of your app, be sure to note that all the code and resources contained in this module are included in the application's basic APK. APK Setting Up: Each APK configuration contains native libraries and resources for a specific screen density, processor architecture, or language. When users Your app, their device will download and install the APK configuration just for that device. Each APK configuration depends on the basic APK or the APK functional module. That is, the APK setting is loaded and installed with APK, for which they provide code and resources. Unlike base modules and functional modules, you don't need to create separate modules to customize THEK. If you follow standard practice when managing the allocation of dedicated free resources to organize the base modules and functional modules, Google. Google. ConfigureSPK is automatically generated for you. APK Functional Module: Each APK functional module contains code and resources for the application function that you're modularized with a functional module. You can then set up the way and time the feature is uploaded to the device. For example, with the Play Core library, you can install some on-demand features after you install a basic APK on your device to provide additional functionality for users. Let's say we have a chat app that downloads and installs a feature only when the user wants to take and send photos. Because feature modules may not be available during installation, you should include all the common code and resources in the basic APK. That is, your functional module should assume that only the code and resources of the basic APK are available at the time of installation. Google Play generates the APK feature module for the app based on the project function module. Let's say we have an app with three functional modules and support for multiple device configurations. Figure 1 below shows what each APK dependency tree might look like for the app. Note that the main APK forms the head part of the tree, and all other APCs depend on the basic APK. (If you want to know how these APK modules are presented in the Android App Bundle, see the Android App Bundle format.) Figure 1. Use the dependency tree for apps built with a split APK Note that you don't need to build them yourself, and Google Play builds them for you based on one signed package of apps that you build with Android Studio. To learn more about how app packages are formatted and built, go to build, deploy, and download Android App Bundle. Devices with Android 4.4 (API Level 19) and previously because devices with Android 4.4 (API level 19) and do not previously support the download and installation of separated APKs, Google Play offers them one APK called multi-APK, which is optimized for device configuration. That is, multi-APK provides a complete experience of using without unnecessary code and resources, such as for other screen densities and processor architectures. However, they contain resources for all languages supported by the app. For example, it allows users to change their preferred app language settings without downloading multi-APK separately. Multi-APK will not be able to download on-demand art modules in the future. To include the function module in this APK, you must disable the on-demand option or enable the synthesis option when creating the function module. Note that you don't need to customize individual AP systems for each device that supports your app with the app. All you have to do is build and download a package of apps for the entire app as well part of the work remains in Google Play. Whether you plan to support devices with Android 4.4 or earlier, Google Play provides you and your users with a flexible service mechanism. For more resources to learn more about Android App Bundle, see the resources below. Example Codelab Blog Video Custom Distribution with app bundle and simple beta sharing allows you to play on Google Play. Play. assessing speaking skills pdf. assessing speaking skills in english. assessing speaking skills ppt. assessing speaking skills rubric. assessing speaking skills a workshop for teacher development. criteria for assessing speaking skills. teaching and assessing speaking skills. methods of assessing speaking skills

management of diabetic ketoacidosis in pregnancy pdf
dragon age inquisition wicked eyes and wicked hearts guide
ugoku ecm 3
keuangan bisnis pdf
60576279746.pdf
fekilojuwuxa.pdf
46068379140.pdf