Article

Preparing an educational assessment app for distribution

Ensure your app maintains academic integrity by reviewing assessment practices and managing system capabilities.

Overview

To help ensure academic integrity for iOS, iPadOS, and macOS assessment apps, it's critical to acquire an Automatic Assessment Configuration (AAC) entitlement, assess your security policies, and manage capabilities. Ensure the AAC entitlement is in your app's provisioning profile and decide how to manage system capabilities not already restricted by AAC, and third-party keyboards. For macOS apps, focus on handling app termination and window operations during assessments to prevent the system from getting locked in assessment mode and providing a poor user experience.

Configure and use the Automatic Assessment Configuration entitlement

Begin by completing the Automatic Assessment Configuration Entitlement Request. After receiving your AAC entitlement, you must set up provisioning before building your app using AAC.

Determine your app's security requirements

AAC creates a secure environment for delivering assessments in iOS, iPadOS, and macOS by restricting hardware and software capabilities at the system level. When a student starts a test in assessment mode in iOS and iPadOS, the device automatically restricts usage to just that specific app. This prevents students from accessing any other apps or capabilities.

In assessment mode, iPadOS and iOS automatically turn off certain capabilities, like the functionality of certain hardware buttons, screen recording and screenshots, remote observation with Classroom, and dictionary definition lookup. Assessment mode also turns off a set of configurable features that you can enable to meet specific assessment needs and requirements. These include:

- Typing capabilities like autocorrect, spell checker, predictive text, keyboard text shortcuts, and smart punctuation, and continuous path keyboard
- Dictation
- Spoken content
- Password autofill
- Handoff

In assessment mode, macOS automatically limits access to one or more apps and restricts access to the following capabilities:

- Mission Control
- Menu bar
- Launching apps using function keys
- Notification Center and notifications
- Apple Music app
- Screen sharing and mirroring
- Screen recording and screenshots
- Siri
- Dictation
- Emoji keyboard
- Trackpad lookup gesture

Important

Consider your app’s assessment security policies to determine what additional capabilities to build into your app.

Restrict third-party software keyboards for iOS and iPadOS

AAC doesn’t restrict third-party software keyboards, and some third-party keyboards support searching from within the keyboard. If you don’t want these used in your assessment app, you restrict their use with `application(_:shouldAllowExtensionPointIdentifier:)` as shown in the code below:

```
// Implement this in the app delegate.
func application(_ application: UIApplication, shouldAllowExtensionPointIdentifier e
    // Disallow custom keyboard extensions.
    if extensionPointIdentifier == .keyboard {
        return false
    }
    return false
}
```

Control the context menu in iOS and iPadOS

A context menu appears when someone selects text or uses a long-press gesture in a text field in iOS or iPadOS. AAC removes the Share, Translate, and Scan Text options from the top-level menu. It’s important to carefully review any additional items that appear and remove them from the menu if you don’t want students to access them during testing.

The function `buildMenu(with:)` removes unwanted items or the entire menu. The following code in the app’s delegate removes the Autofill menu (which gives access to Contacts and Passwords) to prevent people from using Autofill to access content they’ve added:

```
override func buildMenu(with builder: UIMenuBuilder) {
    if #available(iOS 17.0, *) {
        builder.remove(menu: .autoFill)
    } else {
        // Fall back to earlier versions.
    }
}
```

Additionally, the property `lookup` removes the Lookup menu, which is another item in this menu that assessment developers often prefer to remove. For more information about this property, see `UIMenu.Identifier`.

For more information about context menus, see Human Interface Guidelines > Context menus.

Request permissions for your macOS app

AAC prevents the system from showing notifications to people. This means that notifications requesting the user to give permissions, like microphone or camera access, aren't displayed to users. Therefore, it's essential to request all necessary permissions before entering AAC, allowing people to respond appropriately. Additionally, rechecking permissions before entering AAC is advisable to confirm that users have granted all required permissions.

Requesting Authorization for Media Capture on macOS explains how to prompt the user to authorize access to the camera and microphone.

Control the context menu for macOS

You can associate a menu with each view. Your view subclasses can arrange to build their own context menus, or they can customize the default context menu by implementing `menuWillOpen:` in the menu's delegate by calling `removeAllItems`. Another approach is to capture all events at the top level of the application, via `NSEvent.addLocalMonitorForEvents`, and return `nil` for click events so they're ignored, which prevents the appearance of any context menu.

Handle text replacement in macOS

Assessment mode doesn't block text replacements. A person's text replacement dictionary, set in System Settings > Keyboard, is kept in a global default. Your app can override this global default and replace it with an empty dictionary by adding the following:

```
UserDefaults.standard.setValue([:], forKey:"NSUserDictionaryReplacementItems")
```

Replacing with an empty dictionary ensures that your app doesn't offer text replacements in assessment mode. It's important to do this before anything in your UI is loaded, so add it to `init()` of a custom subclass of `NSApplication`.

Prevent early termination of an assessment in macOS

If your app includes a window with an exit button for assessment mode, ensure that people can't close that window while an assessment is ongoing to prevent premature termination of the assessment. One solution is to remove the close and minimize buttons from the window's title bar and the Close menu item if your app has one. You can edit the window's properties in the Attributes inspector or programmatically remove the buttons. You can also arrange to have the window's delegate deny the close operation if a test is in progress.

```swift
// For the window's delegate.
func windowShouldClose(_ sender: NSWindow) -> Bool {
    if aTestIsInProgress {
        return false // Don't allow the window to close.
    }
    return true // It's OK to close the window.
}
```

Even though your app's menu bar isn't visible and someone can't reach it when in assessment mode, people can still type command key equivalents to invoke menu items, such as Command-Q for Quit. Check which of your app's menu items have key equivalents and make sure you handle them properly.

Important

Test these options carefully. Be careful not to lock your development Mac in assessment mode accidentally, which happens when Xcode pauses at a breakpoint while the Mac is in this mode, potentially leaving you with no option but to restart the Mac to exit assessment mode. To prevent such situations, test these functionalities in a virtual machine, which offers the flexibility to force quit if required.

Consider interactions with app-based content filters and VPNs in macOS

AAC restricts network access to every process except those the developer allows. Temporary network access is granted to any process working on behalf of a process that has network access. Some app-based third-party network solutions, such as content filters and VPNs, may prevent an assessment app from connecting to the network during AAC. To accommodate AAC usage on Mac computers with these products, consider these strategies:

- Grant internet access to the network product by including it as a participant in the session.
- Exclude the domains the assessment app requires from being routed through the network product.
- If necessary, schools may temporarily uninstall the incompatible network product. Alternatively, if removal isn't feasible, developers might provide a version of the app that employs different methods to secure the Mac for testing, bypassing the need for AAC.

See Also

Sessions

{} Build an Educational Assessment App

Ensure the academic integrity of your assessment app by using Automatic Assessment Configuration.

`class AEAssessmentConfiguration`

Configuration information for an assessment session.

`class AEAssessmentSession`

A session that your app uses to protect an assessment.