**1. Scripts in the Unity Detection Scene**

1.1 PaintingModuleDataManager

The data core script of the entire app. Data is divided into several core steps: first obtain the latest backend data, generate an XML file from the backend data and save it locally, then compare this XML file with the local resources. Core methods include:

* Start: Initializes data and switches the default language.
* GetXMLFile: Checks if the XML file exists; this file records backend data.
* CreateXML: Creates an XML file.
* SaveXML: Saves an XML file.
* ExistSceneURL: Checks if the scene path exists.
* ExistSceneDownloadURL: Checks if the scene download path exists.
* FirstAddSceneXML: Adds scene resource XML information for the first time.
* FirstDownloadCreateXML: Downloads and creates an XML file for the first time, recording backend room resource information locally.
* FirstAddPMXML: Loads the room's painting resources into the XML file after the first creation of the XML file.
* FirstDownloadRoomExhibits: Downloads the room's painting resources for the first time without comparing with local resources.
* DeleteRoomResources: Deletes resources of a certain room.
* AddRoomResources: Adds resources of a certain room.
* AddRoomPM: Adds the room's painting resources.
* CompareRoomPM: Compares the room's painting data between backend and local.
* ComparisonXMLRoomExhibitsData: Compares local resources with XML data to determine differences and perform additions or deletions.
* LoadXMLComparison: Reads an XML file.
* SceneSelectionLayoutGroupButtonDownloadEvent: Download event for the scene selection button.
* GetRoomDataLocal: Obtains local room information and determines if a download is needed.
* GetRoomPMDataSizeLocal: Gets the file size of the local room's painting resources.
* GetRoomPMDataLocal: Obtains information about the local room's painting data and checks if it's the first download; if not, compares with backend resources.
* ComparisonRoomResourcesXML: Compares local room resources with XML data to detect additions or deletions.
* ComparisonRoomPMXML: Compares local room paintings with XML data to detect additions or deletions.
* GetEnterSceneResources: Gets the resource package to enter the scene.
* GetEnterSceneBGM: Gets the background music file for the scene.
* EnterScenePMInitial: Initializes painting information for the scene.
* LoadLocalXMLToDatas: Loads local XML data into app memory.
* GetPaintingModuleTexture2DResolution: Gets the resolution of the painting module image in the Unity Player scene.
* InitialSceneMask: Initializes the scene mask in the Unity Player scene.
* InitialSceneIntroductionImage: Initializes the introduction image in the scene.
* InitialSceneMainGraph: Initializes the main image in the scene.
* InitialSceneIntroductionText: Initializes the introduction text in the scene.
* InitialSceneAnimationThumbnail: Initializes the thumbnail in the scene.
* SetPaintingModuleMainGraph: Sets the main image of the painting in the Unity Player scene.
* SetPaintingModuleAnimationThumbnail: Sets the frame animation of the painting in the Unity Player scene.
* LoadScenePaintingModules: Loads the scene's painting modules into memory.
* DeleteCurrentPaintingModule: Deletes the current scene's painting module from memory.
* PaintingModuleManagerReplacedTexture: Provides a painting texture in the Unity Player scene.
* CurrentPaintingModuleVideoURL: Gets the URL of the current painting's video in the Unity Player scene.
* DeleteFiles: Deletes files.
* DeleteDirectory: Deletes a folder.
* DetectionPlayerPrefs: Detects account and password stored in memory.
* LogOut: Logs out in the Unity Player scene.
* YoukeRegister: Guest registration in the Unity Player scene.
* ChangePasswordEvent: Changes password.

1.2 HttpManager

The core script for obtaining backend data. Core methods include:

* FirstEnterApp: Checks local data upon the first app entry.
* GetAppVersions: Gets app version information.
* GetAppData: Gets app information.
* JudgeUpdate: Determines if the app needs an update.
* GetWinAppVersionsEvent: Gets Windows version event (not implemented as the app isn't developed for Windows).
* GetAndroidAppVersionsEvent: Gets Android version event.
* GetIosAppVersionsEvent: Gets iOS version event.
* JumpLoginInterface: Navigates to the login interface.
* GetHttpAllGalleryData: Gets all gallery data from the backend.
* GetDataList: Gets data list based on AppID.
* GetGalleryList: Gets gallery list based on AppID.
* GetHttpRoomData: Gets room data.
* GetHttpRoomPMData: Gets all painting data in the room.
* GetHttpRoomPMDataSize: Gets the file size of all painting data in the room.
* GetHttpAllRoomDataSizeEnd: Ends getting the file size of all painting data in the room.
* GetHttpAllRoomData: Gets all room data in the app's galleries.
* GetHttpAllRoomDataEnd: Ends getting all room data in the app's galleries.
* GetHttpAllRoomExhibitsData: Gets all painting data from the backend.
* DetectionDataList: Detects files to download.
* DetectionGalleryList: Detects gallery data.
* DetectionRoomData: Detects room data.
* DetectionRoomPMData: Detects room painting data.
* NetWorkUpdate: Obtains data based on the backend interface.
* PostWebRequest: Sets information for the backend interface transmission.
* GetNetworkState: Gets the current network status.

1.3 PopupWindowCanvasManager

The core script for managing pop-up UI. Core methods include:

* Awake: Initializes and obtains UI components.
* PopupWindow3DOpenEvent: Opens a pop-up based on the PopupWindow3DState (ServerError for backend errors, HttpError for network errors).
* PopupWindow3DCloseEvent: Closes the pop-up.
* SwitchLanguageEvent: Switches the pop-up language based on the current language.
* PopupWindow3DOnClickEvent: Pop-up click event; opens corresponding pop-up based on HttpState.
* UpdateCanvasGroupOpenEvent: Opens the app update pop-up.
* DownloadFileCanvasGroupOpenEvent: Opens the app file download pop-up.
* UpdateCanvasGroupButtonEvent: Navigates to the iOS store.
* DownloadFileCanvasGroupButtonEvent: File download confirmation button event.
* DownloadFileCanvasGroupExitButtonEvent: File download close button event.
* DownloadingExitAccountCanvasGroupButtonEvent: Downloading exit account confirmation button event.
* DownloadingExitAccountCanvasGroupExitButtonEvent: Downloading exit account cancellation button event.
* DownloadingExitAccountCanvasGroupEvent: Downloading exit account UI event.
* ExitAccountCanvasGroupButtonEvent: Exit account confirmation button event.
* ExitAccountCanvasGroupExitButtonEvent: Exit account cancellation button event.
* ExitAccountCanvasGroupEvent: Exit account UI event.
* DownloadSuccessCanvasGroupButtonEvent: Download success confirmation button event.
* DownloadSuccessCanvasGroupExitButtonEvent: Download success cancellation button event.
* DownloadSuccessCanvasGroupEvent: Download success UI event.

**2. Scripts in the Unity Main Scene**

2.1 Downloader

The download manager. Core methods include:

* OnGetFileSize: After obtaining the file size, adds it to the downloader.
* DownloadCheck: Performs download checks.
* DownloaderSwitchRoomsResetData: Resets data after the downloader switches rooms.

2.2 AccountManager

Manages the UI and interaction logic for the Unity Main scene. Core methods include:

* Awake: Obtains UI components and adds UI button events.
* Start: Initializes some data.
* SwitchLanguageEvent: Switches language event.
* AccountSigninButtonDengluEvent: Home page login button event (includes account detection and sending data to the backend).
* AccountSigninButtonWangjimimaEvent: Home page forgot password button event.
* AccountSigninButtonJizhumimaEvent: Home page remember password button event.
* AccountSigninButtonPasswordInputFieldButtonEvent: Home page login hide password event.
* AccountSigninButtonZhuceEvent: Home page register button event.
* AccountSigninButtonYoukeEvent: Home page guest login event.
* LoginSuccessEvent: Login success event.
* AccountRegisterButtonFasongEvent: Register send code button event.
* AccountRegisterButtonFasongEventSuccess: Register send button success event.
* AccountRegisterButtonChongxinResetEvent: Register send button reset event.
* AccountRegisterButtonChongxinEvent: Register resend code event.
* AccountRegisterButtonQuedingEvent: Register confirmation button event.
* AccountRegisterButtonReturnEvent: Register return button event.
* RegisterSuccessEvent: Register success event.
* AccountRegisterButtonShurumimaInputFieldButtonEvent: Register hide password event.
* AccountRegisterButtonZaicimimaInputFieldButtonEvent: Register hide re-enter password event.
* ChangePasswordButtonFasongEvent: Forgot password send button event.
* ChangePasswordButtonFasongEventSuccess: Forgot password send button success event.
* ChangePasswordButtonChongxinResetEvent: Forgot password resend button reset event.
* ChangePasswordButtonChongxinEvent: Forgot password resend event.
* ChangePasswordButtonQuedingEvent: Forgot password confirmation button event.
* ChangePasswordButtonReturnEvent: Forgot password return button event.
* ChangePasswordButtonShurumimaInputFieldButtonEvent: Change password hide password event.
* ChangePasswordButtonZaicimimaInputFieldButtonEvent: Change password hide re-enter password event.
* ConsentClauseButtonNextEvent: Consent clause next button event.
* ConsentClauseButtonReturnEvent: Consent clause return button event.
* ConsentClauseButtonAgreeEvent: Consent clause agree button event.
* SceneSelectionButtonReturnEvent: Scene selection return button event.
* SceneSelectionButtonReturnCanvasGroupEvent: Scene selection return button UI event.
* SceneSelectionButtonNextEvent: Scene selection next button event.
* SceneSelectionLayoutGroupButtonEvent: Scene selection confirmation event.
* CalculatePlaces: Counts characters.

**3. Scripts in the Unity Player Scene**

3.1 LoadSceneManager

Loads the scene resource package. Core methods include:

* LoadSelectScene: Obtains information about the selected scene.
* ILoadAddScene: Loads the resource package for the current scene.
* UninstallcurrentSceneAB: Uninstalls the current scene resource package.

3.2 DataManager

Manages data for the current scene. Core methods include:

* IInitial: Initializes scene data (background music, painting UI, minimap, frame animation, and joystick control).
* StartLoadSceneEvent: Starts the scene loading event.
* FinishLoadSceneEvent: Ends the scene loading event.

3.3 EasyTouchControlsManager

Manages the joystick. Core methods include:

* Initial: Initializes joystick control.
* Reset: Resets joystick control.
* EasyTouchControlsStart: Starts joystick control.
* EasyTouchControlsStop: Stops joystick control.
* PlayerTransferMap: Transfers the player to a map location.
* SetETCJoystickSpeed: Sets joystick speed.
* SetETCTouchPadSpeed: Sets screen sliding speed.

3.4 PaintingModuleFrameCanvas

Manages painting UI and interactions. Core methods include:

* Initial: Initializes painting UI.
* Reset: Resets painting UI.
* OptionButtonInitial: Initializes painting UI buttons.
* PaintingModuleFrameStart: Starts painting frame animation.
* PaintingModuleFrameStop: Stops painting frame animation.
* PaintingModuleVideoStart: Starts playing painting video.
* PaintingModuleVideoStop: Stops playing painting video.
* PaintingModuleIntroduceStart: Opens painting introduction.
* PaintingModuleIntroduceStop: Closes painting introduction.
* PaintingModuleWebStart: Opens painting web.
* PaintingModuleWebStop: Closes painting web.
* PaintingModuleSoundStart: Opens painting audio.
* PaintingModuleSoundStop: Closes painting audio.
* PaintingModuleStoreStart: Opens painting store.
* PaintingModuleWebVideoStart: Opens painting web video.
* PaintingModuleLoadingStart: Opens painting loading.
* PaintingModuleLoadingStop: Closes painting loading.
* PaintingModuleLoadingLoad: Loads painting loading.
* PaintingModuleFrameRawImageReplace: Switches painting frame images.
* PaintingModuleFrameRawSetAnimationThumbnail: Sets painting thumbnail.
* PaintingModuleFrameRawImageLocalScale: Adjusts painting frame image size.
* PaintingModuleIntroduceRawImageLocalScale: Adjusts painting introduction image size.
* PaintingModuleWebButtonEventAdd: Adds web button events for paintings.
* PaintingModuleWebButtonEventExit: Exits web button events for paintings.
* PaintingModuleIntroduceButtonEventAdd: Adds introduction button events for paintings.
* SetPaintingModuleIntroduce: Sets painting introduction.
* PaintingModuleSoundButtonEventAdd: Adds audio button events for paintings.
* SetPaintingModuleOptionButton: Sets painting button switches.
* PaintingModuleFrameLastButtonEvent: Previous painting button event.
* PaintingModuleFrameNextButtonEvent: Next painting button event.
* PaintingModuleLockExit: Painting lock exit.
* PaintingModuleStoreButtonEventAdd: Adds store button events for paintings.
* PaintingModuleWebVideoButtonEventAdd: Adds web video button events for paintings.

3.5 PaintingModuleFrameControl

Controls painting frame animation, using different states in the Update method. Core methods include:

* PaintingModuleFrameControlStart: Starts painting frame animation control.
* PaintingModuleFrameControlStop: Stops painting frame animation control.
* PaintingModuleDistanceInitial: Initializes painting position and distance.
* PaintingModuleLocalPositionInitial: Initializes painting position.
* PaintingModuleIntroduceTextureInitial: Initializes painting introduction position.

3.6 PaintingModule

Manages each painting. Core methods include:

* Awake: Initializes data and adds UI events.
* ColliderEnterEvent: Collider start event.
* ColliderExitEvent: Collider end event.
* ClickColliderExitEvent: Collider end method after clicking.
* DeletePaintingModuleTextures: Clears painting image groups.

3.7 PaintingModuleMap

Manages the painting scene map and settings. Core methods include:

* Initial: Initializes map data and adds button events.
* PaintingModuleMapUICanvasStart: Opens the painting map UI.
* PaintingModuleMapUICanvasExit: Closes the painting map UI.
* PaintingModuleHelpUICanvasStart: Opens the painting help UI.
* PaintingModuleHelpUICanvasExit: Closes the painting help UI.
* PaintingModuleMapCanvasNormalMode: Opens the painting map in Normal mode.
* PaintingModuleMapCanvasSamrtMode: Opens the painting map in Smart mode.
* PaintingModuleMapCanvasSettingEvent: Painting map settings event.
* PaintingModuleMapCanvasStart: Opens the painting map.
* PaintingModuleMapCanvasExit: Closes the painting map.
* PaintingModuleHelpCanvasStart: Opens painting help.
* PaintingModuleHelpCanvasExit: Closes painting help.
* LoadMapAnimationThumbnailTexture: Loads map thumbnail.
* PaintingModuleSettingCanvasStart: Opens painting settings.
* PaintingModuleSettingCanvasExit: Closes painting settings.
* PaintingModuleSettingCanvasSlider\_BGMMuteLogoButtonOnEvent: Unmutes painting audio.
* PaintingModuleSettingCanvasSlider\_BGMMuteLogoButtonOffEvent: Mutes painting audio.
* PaintingModuleSettingCanvas\_CebianlanShezhiButtonEvent: Painting settings sidebar button event.
* PaintingModuleSettingCanvas\_CebianlanZhanghucaozuoButtonEvent: Account operations sidebar button event.
* PaintingModuleSettingCanvas\_CebianlanFukuanguanliButtonEvent: Payment management sidebar button event.
* PaintingModuleSettingCanvas\_CebianlanGuanyuyingyongButtonEvent: About app sidebar button event.
* PaintingModuleSettingCanvas\_DeleteAccountEvent: Delete account event.
* PaintingModuleSettingCanvas\_DeleteAccountPopupButtonYesEvent: Confirm delete account in popup.
* PaintingModuleSettingCanvas\_DeleteAccountPopupButtonNoEvent: Cancel delete account in popup.

3.8 SceneBGMManager

Manages background music for the painting scene. Core methods include:

* GetSceneBGM: Obtains and loads local background music files.
* PlaySceneBGM: Plays scene background music.
* PlayIntroduceSceneBGM: Plays introduction background music.
* PauseSceneBGM: Pauses scene background music.
* UnPauseSceneBGM: Resumes scene background music.
* MuteSceneBGM: Mutes or unmutes scene background music.

3.9 PaintingModuleLockManager

Manages locked paintings. Core methods include:

* **Awake**: Obtains UI components and adds UI button events.
* **Start**: Initializes properties.
* **PaintingModuleLockOpenEvent**: Opens painting lock.
* **PaintingModuleLockCloseEvent**: Closes painting lock.
* **PaintingModuleLockExitButtonEvent**: Painting lock exit event.
* **PaintingModuleLockRegisterButtonEvent**: Painting lock register navigation event.
* **PaintingModuleLockChangePasswordButtonEvent**: Painting lock password change navigation event.
* **AccountRegisterButtonShurumimaInputFieldButtonEvent**: Register password hide button event.
* **AccountRegisterButtonZaicimimaInputFieldButtonEvent**: Register re-enter password hide button event.
* **AccountRegisterButtonFasongEvent**: Register send button event.
* **AccountRegisterButtonQuedingEvent**: Register confirm button event.
* **AccountRegisterButtonReturnEvent**: Register return button event.
* **AccountRegisterButtonChongxinEvent**: Register resend code event.
* **AccountRegisterButtonChongxinResetEvent**: Register resend button reset event.
* **AccountRegisterButtonFasongEventSuccess**: Register send button success event.
* **RegisterSuccessEvent**: Register success event.
* **PopupWindowOpenEvent**: Opens popup.
* **PopupWindowCloseEvent**: Closes popup.
* **PopupWindowZidingyiEvent**: Custom popup event.
* **CalculatePlaces**: Determines characters.

3.10 PaintingModuleMask

Cancels the bottom mask of the painting UI; the mask disappears after loading to prevent accidental touches before entering the scene.

**4. Other Scripts**

4.1 DataJsonClass

Records backend interface data; the first line is the test server address, the second is the production server address. Use the needed one and comment out the other.

4.2 PaintingModuleCollider

Painting collider detection script, added to painting colliders by default. Core methods include:

* **OnTriggerEnter**: Collider trigger start event.
* **OnTriggerExit**: Collider trigger end event.

4.3 CanHideInputField

Dynamically refreshes to hide or show passwords.

4.4 Messenger

Event manager.

4.5 TimeMgr

Timer for time-based events.

4.6 UnitySingleton

Singleton pattern implementation.