```swift
//
//  PlayMoveRESTWebAPIModelAccessStrategy.swift
//  f30
//
//  Created by David on 05/02/2018.
//  Copyright © 2018 com.smartfoundation. All rights reserved.
//

import SFCore
import SFModel
import SFSocial
import SFSerialization
import SFNet
import f30Core
import f30Model

/// A strategy for accessing the PlayMove model data using a REST Web API
public class PlayMoveRESTWebAPIModelAccessStrategy:
 RESTWebAPIModelAccessStrategyBase {

    // MARK: - Initializers

    private override init() {
        super.init()
    }

    public override init(connectionString: String,
                         storageDateFormatter: DateFormatter) {
        super.init(connectionString: connectionString,
                   storageDateFormatter: storageDateFormatter,
                   tableName: "PlayMoves")

    }


    // MARK: - Private Methods

    fileprivate func runQuery(byPlayTileID playTileID: String, playGameID: String,
     into collection: ProtocolModelItemCollection, oncomplete
     completionHandler:@escaping ([String:Any]?, Error?) -> Void) {

        #if DEBUG

            if (ApplicationFlags.flag(key: "LoadPlayMovesDummyDataYN")) {

                self.selectDummy(byPlayTileID: playTileID, playGameID: playGameID,
                 into: collection, oncomplete: completionHandler)

                return

            }

        #endif

        // Create the dataWrapper
        let dataWrapper:            DataJSONWrapper = DataJSONWrapper()
    // Create processResponse completion handler
    let processResponseCompletionHandler: (([String:Any]?, URLResponse?,
     Error?) -> Void) =
    {
        (data, response, error) -> Void in  // [weak self]

        // Call the completion handler
        completionHandler(data, error)
    }

    // Create processResponse
    let processResponse:        ((NSMutableData?, URLResponse?, Error?) ->
     Void) = self.getProcessResponse(oncomplete:
     processResponseCompletionHandler)

    // Create restApiHelper
    let restApiHelper:          RESTApiHelper = RESTApiHelper(processResponse:
     processResponse, mode: RESTApiHelperMode.CompletionHandler)

    // Get the Url
    var urlString:              String =
     NSLocalizedString("PlayMovesSelectByPlayTileIDUrl", tableName:
     "RESTWebAPIConfig", comment: "")
    urlString                   = String(format: urlString,
                                          playTileID,
                                          playGameID)

    // Call the REST Api
    restApiHelper.call(urlString: urlString, httpMethod: .POST, data:
     dataWrapper)

}

fileprivate func runQuery(byPlayTokenID playTokenID: String, playGameID:
 String, into collection: ProtocolModelItemCollection, oncomplete
 completionHandler:@escaping ([String:Any]?, Error?) -> Void) {

    #if DEBUG

        if (ApplicationFlags.flag(key: "LoadPlayMovesDummyDataYN")) {

            self.selectDummy(byPlayTokenID: playTokenID, playGameID:
             playGameID, into: collection, oncomplete: completionHandler)

            return

        }

    #endif

    // Create the dataWrapper
    let dataWrapper:            DataJSONWrapper = DataJSONWrapper()

    // Create processResponse completion handler
    let processResponseCompletionHandler: (([String:Any]?, URLResponse?,
     Error?) -> Void) =
    {
            (data, response, error) -> Void in  // [weak self]

            // Call the completion handler
            completionHandler(data, error)
        }

        // Create processResponse
        let processResponse:        ((NSMutableData?, URLResponse?, Error?) ->
         Void) = self.getProcessResponse(oncomplete:
         processResponseCompletionHandler)

        // Create restApiHelper
        let restApiHelper:          RESTApiHelper = RESTApiHelper(processResponse:
         processResponse, mode: RESTApiHelperMode.CompletionHandler)

        // Get the Url
        var urlString:              String =
         NSLocalizedString("PlayMovesSelectByPlayTokenIDUrl", tableName:
         "RESTWebAPIConfig", comment: "")
        urlString                   = String(format: urlString,
                                             playTokenID,
                                             playGameID)

        // Call the REST Api
        restApiHelper.call(urlString: urlString, httpMethod: .POST, data:
         dataWrapper)

    }

    fileprivate func runQuery(byPlayTokenID playTokenID: String, playGameID:
     String, playAreaPathData: String, into collection:
     ProtocolModelItemCollection, oncomplete completionHandler:@escaping
     ([String:Any]?, Error?) -> Void) {

        #if DEBUG

            if (ApplicationFlags.flag(key: "LoadPlayMovesDummyDataYN")) {

                self.selectDummy(byPlayTokenID: playTokenID, playGameID:
                 playGameID, playAreaPathData: playAreaPathData, into: collection,
                 oncomplete: completionHandler)

                return

            }

        #endif

        // Create the dataWrapper
        let dataWrapper:            DataJSONWrapper = DataJSONWrapper()
        dataWrapper.setParameterValue(key: "\
         (PlayMoveDataParameterKeys.PlayAreaPathData)", value: playAreaPathData)

        // Create processResponse completion handler
        let processResponseCompletionHandler: (([String:Any]?, URLResponse?,
         Error?) -> Void) =
        {
            (data, response, error) -> Void in  // [weak self]

            // Call the completion handler
            completionHandler(data, error)
        }

        // Create processResponse
        let processResponse:        ((NSMutableData?, URLResponse?, Error?) ->
         Void) = self.getProcessResponse(oncomplete:
         processResponseCompletionHandler)

        // Create restApiHelper
        let restApiHelper:          RESTApiHelper = RESTApiHelper(processResponse:
         processResponse, mode: RESTApiHelperMode.CompletionHandler)

        // Get the Url
        var urlString:              String =
         NSLocalizedString("PlayMovesSelectByPlayTokenIDPlayAreaPathUrl",
         tableName: "RESTWebAPIConfig", comment: "")
        urlString                   = String(format: urlString,
                                               playTokenID,
                                               playGameID)

        // Call the REST Api
        restApiHelper.call(urlString: urlString, httpMethod: .POST, data:
         dataWrapper)

    }



    // MARK: - Override Methods


    // MARK: - Dummy Data Methods

    fileprivate func selectDummy(byPlayTileID playTileID: String, playGameID:
     String, into collection: ProtocolModelItemCollection, oncomplete
     completionHandler:@escaping ([String:Any]?, Error?) -> Void) {

        let defaultKey:         String = "byPlayTileID"
        let key:                String = defaultKey + "_\(playTileID)"

        // Get string for key with playTileID
        var responseString:     String  = NSLocalizedString(key, tableName:
         "PlayMovesDummyRESTWebAPIResponse", comment: "")

        // If not found then use defaultKey
        if (responseString == key) {

            responseString      = NSLocalizedString(defaultKey, tableName:
             "PlayMovesDummyRESTWebAPIResponse", comment: "")

        }

        // Convert the response to JSON dictionary
        let data:               [String:Any]? =
         JSONHelper.stringToJSON(jsonString: responseString) as? [String:Any]

        // Process the data
        let returnData:         [String:Any]? =
         self.processRESTWebAPIResponse(responseData: data!)

        // Call the completion handler
        completionHandler(returnData, nil)

    }

    fileprivate func selectDummy(byPlayTokenID playTokenID: String, playGameID:
     String, into collection: ProtocolModelItemCollection, oncomplete
     completionHandler:@escaping ([String:Any]?, Error?) -> Void) {

        let responseString  = NSLocalizedString("byPlayTokenID", tableName:
         "PlayMovesDummyRESTWebAPIResponse", comment: "")

        // Convert the response to JSON dictionary
        let data:           [String:Any]? = JSONHelper.stringToJSON(jsonString:
         responseString) as? [String:Any]

        // Process the data
        let returnData:     [String:Any]? =
         self.processRESTWebAPIResponse(responseData: data!)

        // Call the completion handler
        completionHandler(returnData, nil)

    }

    fileprivate func selectDummy(byPlayTokenID playTokenID: String, playGameID:
     String, playAreaPathData: String, into collection:
     ProtocolModelItemCollection, oncomplete completionHandler:@escaping
     ([String:Any]?, Error?) -> Void) {

        let responseString  = NSLocalizedString("byPlayTokenIDPlayAreaPath",
         tableName: "PlayMovesDummyRESTWebAPIResponse", comment: "")

        // Convert the response to JSON dictionary
        let data:           [String:Any]? = JSONHelper.stringToJSON(jsonString:
         responseString) as? [String:Any]

        // Process the data
        let returnData:     [String:Any]? =
         self.processRESTWebAPIResponse(responseData: data!)

        // Call the completion handler
        completionHandler(returnData, nil)

    }

}

// MARK: - Extension ProtocolPlayMoveModelAccessStrategy
```

```
extension PlayMoveRESTWebAPIModelAccessStrategy:
 ProtocolPlayMoveModelAccessStrategy {

    // MARK: - Public Methods

    public func select(byPlayTileID playTileID: String, playGameID: String,
     collection: ProtocolModelItemCollection, oncomplete
     completionHandler:@escaping ([String:Any]?, ProtocolModelItemCollection?,
     Error?) -> Void) {

        // Create completion handler
        let runQueryCompletionHandler: (([String:Any]?, Error?) -> Void) =
         self.getRunQueryCompletionHandler(collection: collection, oncomplete:
         completionHandler)

        // Run the query
        self.runQuery(byPlayTileID: playTileID, playGameID: playGameID, into:
         collection, oncomplete: runQueryCompletionHandler)

    }

    public func select(byPlayTokenID playTokenID: String, playGameID: String,
     collection: ProtocolModelItemCollection, oncomplete
     completionHandler:@escaping ([String:Any]?, ProtocolModelItemCollection?,
     Error?) -> Void) {

        // Create completion handler
        let runQueryCompletionHandler: (([String:Any]?, Error?) -> Void) =
         self.getRunQueryCompletionHandler(collection: collection, oncomplete:
         completionHandler)

        // Run the query
        self.runQuery(byPlayTokenID: playTokenID, playGameID: playGameID, into:
         collection, oncomplete: runQueryCompletionHandler)

    }

    public func select(byPlayTokenID playTokenID: String, playGameID: String,
     playAreaPathData: String, collection: ProtocolModelItemCollection, oncomplete
     completionHandler:@escaping ([String:Any]?, ProtocolModelItemCollection?,
     Error?) -> Void) {

        // Create completion handler
        let runQueryCompletionHandler: (([String:Any]?, Error?) -> Void) =
         self.getRunQueryCompletionHandler(collection: collection, oncomplete:
         completionHandler)

        // Run the query
        self.runQuery(byPlayTokenID: playTokenID, playGameID: playGameID,
         playAreaPathData: playAreaPathData, into: collection, oncomplete:
         runQueryCompletionHandler)

    }

}
```