AIW 基準(Criteria) API

Note: :このAPIは、SDKの標準には含まれていません。SDK API Plug-inに含まれています。

AIW 基準API は、AIW基準アイテムとインタレスト(interest)を利用するための手儀とクラスを提供します。基準アイテムは、コンシューマ(consumer)アプリケーションが使いたいAIWサービスを定義しており、インタレストは、基準アイテムの配列です。一方、AIW サービスは、AIWサービス・プロバイダによって提供されます。

利用例

最も重要なAIW基準APIの使用例として次のものがあります:

リソース・ファイル内でコンシューマのインタレストを定義

コンシューマのインタレストの動的定義

非同期のサービス呼び出し

サンプル・コード

リソース・ファイル内でコンシューマのインタレストを定義

通常、コンシューマのインタレストは、リソース・ファイル内に定義されます。以下の例題では、メニューとベース・サービスのインタレストを含みます:

```
#include <AiwCommon.rh>
```

```
RESOURCE AIW_INTEREST r_aiwexample_menuinterest
    {
    items =
        {
        AIW_CRITERIA_ITEM
            {
            id = EAIWExampleHelpPlaceholder;
            serviceCmd = KAiwCmdHelp;
            contentType = "*";
            serviceClass = KAiwClassMenu;
            }
        };
    }
```

```
RESOURCE AIW_INTEREST r_aiwexample_baseinterest
    {
    items =
        {
        AIW_CRITERIA_ITEM
            {
            id = EAIWExampleBaseServiceId;
            serviceCmd = KAiwCmdMnShowMap;
            contentType = "application/x-landmark";
            serviceClass = KAiwClassBase;
            }
        };
    }
```

この例題では、各インタレストに付き一つの基準アイテムだけが定義されています。しかし、コンマ区切りで、複数個も可能です。

最初の基準アイテムは、メニュー・サービスを参照しています。すなわち、対応するAIWプロバイダは、コンシューマ・アプリケーションのメニューに、メニュー・アイテムを追加することになっています。2番目の基準アイテムは、ベースサービスを参照しています。ベース・サービス・コマンドの使用は、メニューを要求しません。id 列挙は、コンシューマ・アプリケーションのHRHファイルに定義されなければいけません。

メニュー・サービスのためのメニュー・アイテムは、MENU_PANE リソースで定義されます。例えば次のようになります:

```
RESOURCE MENU_PANE r_aiwexample_menu
    {
    items =
        {
        MENU_ITEM
            {
            command = EAIWExampleHelpPlaceholder;
            txt = "Help submenu";
            cascade = AIW_INTELLIGENT_CASCADE_ID | AIW_LOCK_SUBMENU_TITLE;
            },
        MENU_ITEM
            {
            command = EAknSoftkeyExit;
            txt = "Exit";
            }
```


(C) Copyright Nokia 2013. All rights reserved.

```
        };
    }
```

この例では、最初のアイテムは、AIWメニュー・アイテムで、2番目は通常のメニュー・アイテムです。AIW_INTELLIGENT_CASCADE_ID は、プロバイダーが複数のAIWメニュー・アイテムを提供する場合は、AIWフレームワークは、AIWメニュー・アイテムをサブメニューに置きます。もし、利用可能なメニュー・アイテムが一つだけならば、それはメイン・レベルにおかれます。

AIW_LOCK_SUBMENU_TITLE は、もしサブメニューがあった場合、AIWフレームワークは、コンシューマが定義した文字列"Help submenu"を使用します。つまり、もし複数のプロバイダがサブメニュー・タイトルを提供していても、それは無視されます。

コンシューマのインタレストの動的定義

インタレストは、動的に定義することも出来ます。次の例があります:

```
// Create AIW service handler
CAiwServiceHandler* serviceHandler = CAiwServiceHandler::NewLC();

// Create AIW interest
RCriteriaArray interest;
CleanupClosePushL(interest);
_LIT8(KContentTypeLandmark, "application/x-landmark");
CAiwCriteriaItem* criteria = CAiwCriteriaItem::NewLC(KAiwCmdMnShowMap,
                                KAiwCmdMnShowMap, KContentTypeLandmark);
// We are using a base service.
TUid base;
base.iUid = KAiwClassBase;
criteria->SetServiceClass(base);

User::LeaveIfError(interest.Append(criteria));

// Attach the interest to the AIW framework.
serviceHandler->AttachL(interest);

...
// Execute AIW commands etc.
...

// Pop and destroy when not any more needed.
CleanupStack::PopAndDestroy(3); // criteria, interest, servicehandler
```

この例では、定数KAiwCmdMnShowMap は、基準アイテムidとしても使用されています。このような方法も可能です。コンシューマは、使用するidを決めることが出来ます。(そして、ベース・サービスでは、この基準アイテムidは全く意味がありません。)

非同期のサービス呼び出し

いくつかのAIWサービス・プロバイダーは、非同期サービス呼び出しをサポートします。つまり、処理に時間がかかり、プロバイダーがすぐにリターンできない場合です。

このような場合、コンシューマーはMAiwNotifyCallback クラスから派生し、MAiwNotifyCallback::HandleNotifyL() メソッドを実装しなければいけません。コンシューマがCAiwServiceHandler::ExecuteMenuCmdL() か CAiwServiceHandler::ExecuteServiceCmdL() を呼ぶ時、M クラスへのコールバック・ポインタを設定し、aCmdOptions にKAiwOptASyncronous を設定します。サービス・プロバイダは、終了時に、コールバック・メソッドをKAiwEventCompleted 付きで呼びます。利用可能なイベント・コードのリストは、AiwCommon.hrh を参照してください。

MAiwNotifyCallback からの派生の方法の例が、次になります:

```
#include <AiwCommon.rh>
```

```
class CMyConsumerApp : public MAiwNotifyCallback
    {
    ...
    public:
        // From MAiwNotifyCallback
        virtual TInt HandleNotifyL(
            TInt aCmdId,
            TInt aEventId,
            CAiwGenericParamList& aEventParamList,
            const CAiwGenericParamList& aInParamList);
    ...
```

HandleNotifyL() の実装は、次のようになります:

```
TInt CMyConsumerApp::HandleNotifyL(
    TInt /*aCmdId*/,
    TInt aEventId,
    CAiwGenericParamList& /*aEventParamList*/,
    const CAiwGenericParamList& /*aInParamList*/)
    {
    // Service command (aCmdId) can be checked here, if necessary.

    // Check the event code.
    switch (aEventId)
        {
        case KAiwEventCanceled:
            {
            // Handle cancellation...
            break;
            }
        case KAiwEventCompleted:
            {
            // Handle completion...
            break;
            }
        ...
        // Other event codes can be handled here as well.
        ...
        default:
```


(C) Copyright Nokia 2013. All rights reserved.

```
            break;
        }

    return KErrNone;
    }
```

非同期コマンド処理へのリクエストは次のようになります:

```
void CMyConsumerApp::HandleCommandL(TInt aCommand)
    {
    ...
    // Execute AIW menu service command asynchronously. Remember to check from
    // the provider documentation whether it supports asynchronous command handling.
    // No input and output paramters are used in this example. Check from the
    // provider documentation whether those are required.
    iServiceHandler->ExecuteMenuCmdL(
        aCommand,                           // The menu command
        iServiceHandler->InParamListL(),    // No input parameters used
        iServiceHandler->OutParamListL(),   // No output parameters used
        KAiwOptASyncronous,                 // Asynchronous command handling requested
        this );                             // MAiwNotifyCallback pointer
    ...
```

例題プロジェクト

File:AIWConsumerBasics.zip

Known issues

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.