帰ってきたC++

CommunityOpenDay2013
2013/5/11 Sat

Room metro大阪 遥佐保

はじめに

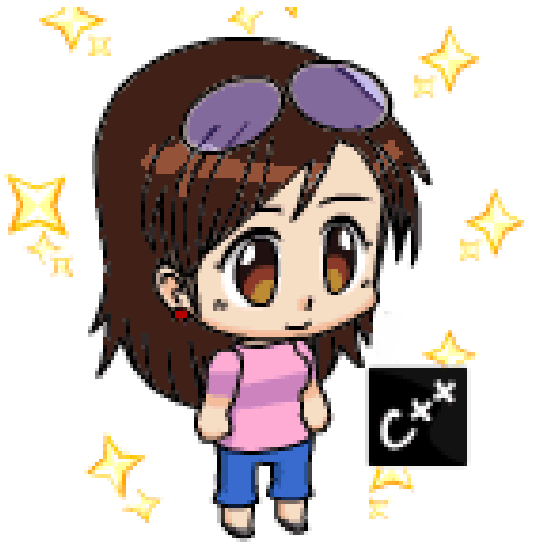
自己紹介

@hr_sao

名前と経歴を言う

出没コミュニティ

- Room metro（大阪）
- ExceptionalC++読書会（大阪）
- SQLWorld （大阪）



本日の目的

VisualStudio2012の
C++関連の
理解度を深める

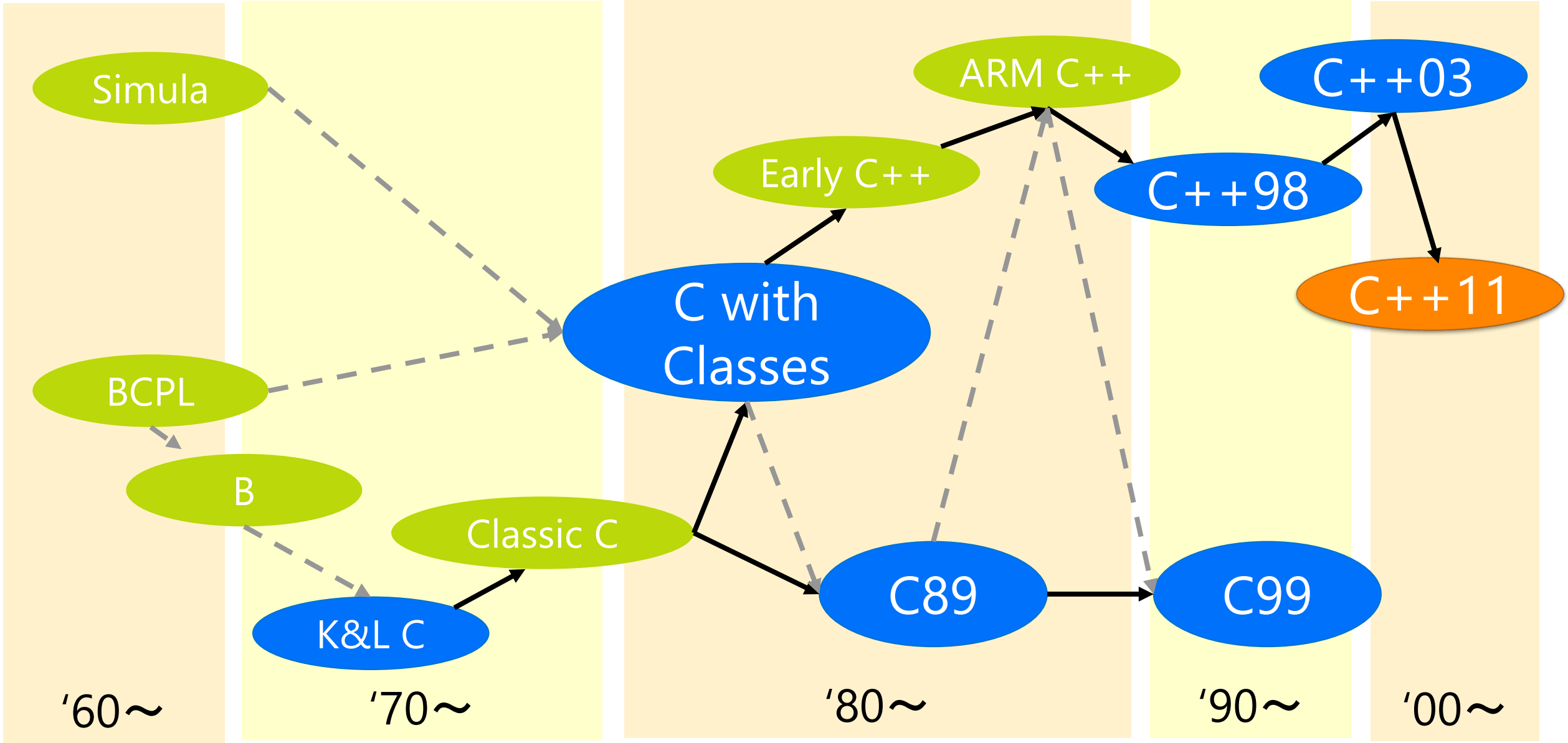
Topics

0. C++とその周りの歴史
1. C++ separate programing categories
2. Windows application – UI
3. How can you use Visual Studio 2012?
4. Windows store apps – C++/CX
5. How to learn C++/CX

-. Welcome back to Windows Phone

0. C++とその周りの歴史

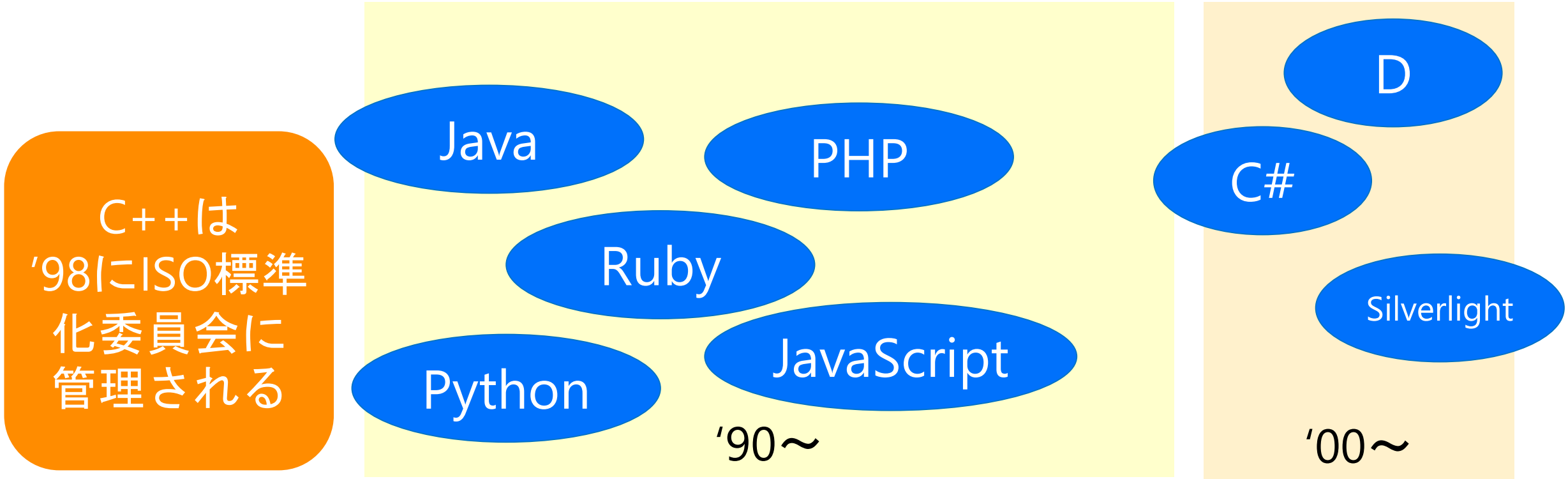
C/C++の歴史


Simula
BCPL
B
Classic C
K&L C
C with Classes
Early C++
ARM C++
C++98
C++03
C++11
C89
C99
'60~
'70~
'80~
'90~
'00~

90年代以降をよく見て見る

より上位の高級言語を目指したJavaとC#

Web系言語の登場　　...>どうみても押され気味?



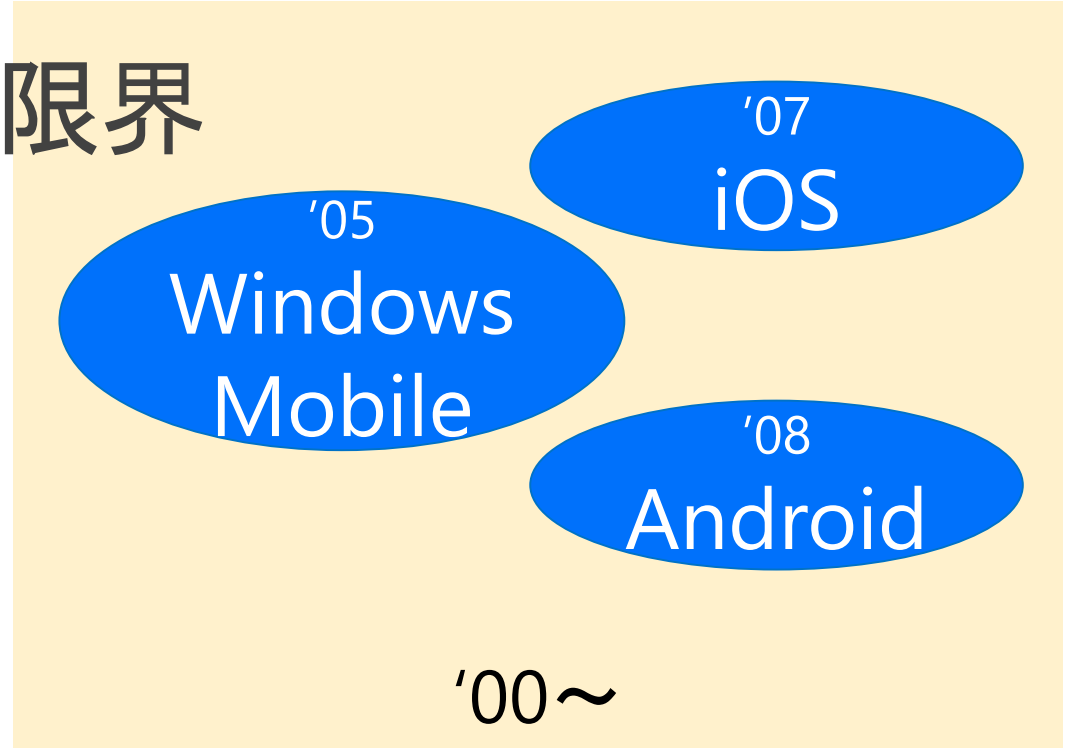
2000年後半にスマートフォンの登場

iOSとAndroidが登場

- ナローバンドでの利用前提
- Web画面遷移の操作感の限界

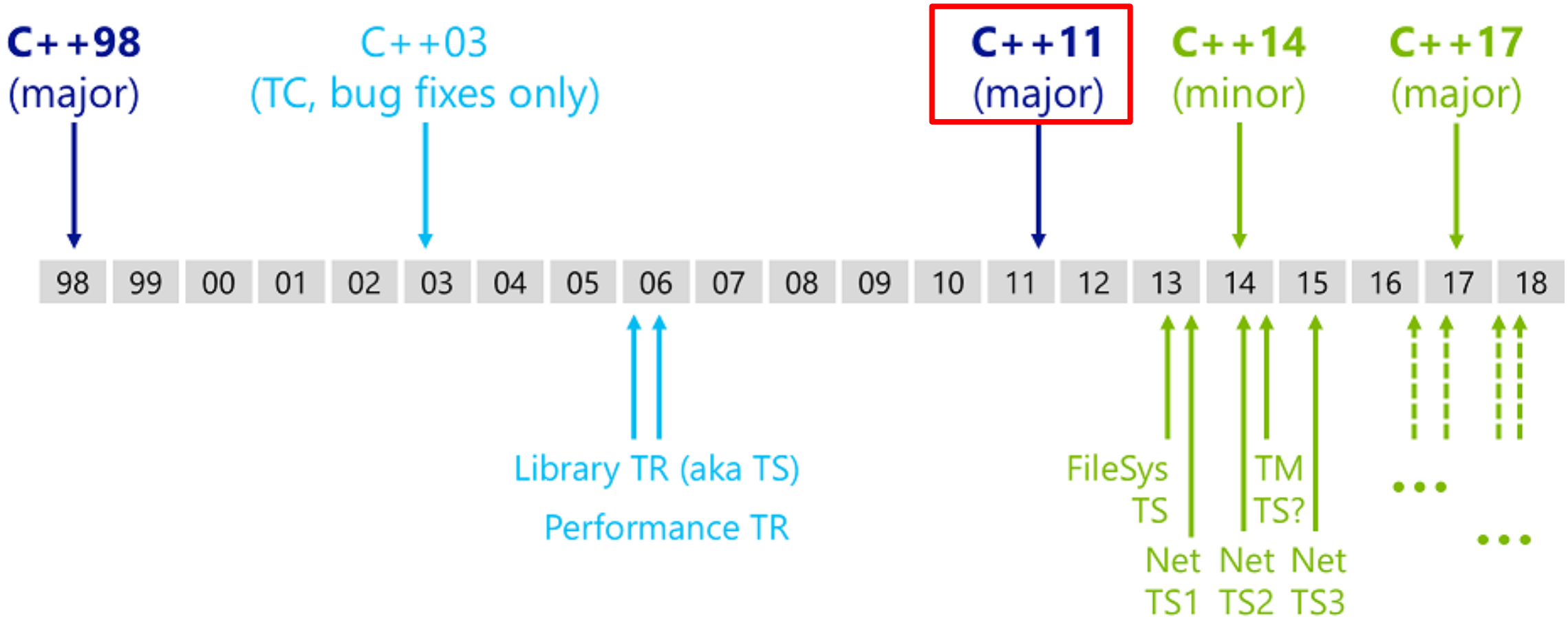
クライアント側に情報を！

UIとロジックの分離（流行った）
とにかく操作感良く！



Recently Published: C++11 (2011)

http://isocpp.org/std/status

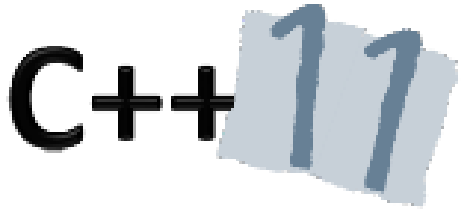


Welcome back to C++ ☺

いったんWebに走ったみんなの気持ちが

クライアントサイド(ネイティブ)に戻ってきた…！

C++11規格をひっさげて



VisualStudio2012が

通りますよっ

Now... use cases for C++

1. Portability

   LinuxとWindowsで同じコードを動かす

2. Reusability

   既存ライブラリ使いたい

3. Performance

   .NETがリッチすぎる

1. C++ separate programing categories

C++の世界

プログラミングのカテゴリが3つある

- C言語を使う
- Classを使う
- Templateを使う

C++ Primer Plus (6th Edition) http://www.amazon.com/Primer-Plus-6th-Developers-Library/dp/0321776402

Day 1 Keynote - Bjarne Stroustrup: C++11 Style http://channel9.msdn.com/Events/GoingNative/GoingNative-2012/Keynote-Bjarne-Stroustrup-Cpp11-Style

C言語の世界

手続き型言語

- 開発効率が高いとは言えないがシンプル
- 実行速度が速く、サイズも小さい

↑↑↑もちろん作り方に依存するが、普通はそうなる

functionとstructの世界

macros / void* / casts の多用スキルが身についてしまう

classの世界

C言語にプラスされたオブジェクト指向言語

structにもっと機能を入れたなどなど

private / operator overload / inheritance / interface / exception / virtual functions …

- C言語よりは開発効率は上がる

↑↑↑もちろん作り方に依存するが、普通はそうなる

C++85 style (C with Classes)

templateの世界

ジェネリック/メタプログラミングが可能

データの型には依存せず

任意Objectを格納できるコンテナ

クラス・関数テンプレート / 特殊化 / テンプレートパラメータ…

- 型・数値の差し替え、型推論でコーディング量の減少が可能に
- その他応用例も多数

いつでもどこでも
Template(^^)

sample: C style and C++11

```
// for-loop
sum = 0;
for( vector<int>size_type i = 0; i < v.size(); i ++ ){
        sum += v [ i ];
}
```

```
// for_each + lambda
sum = 0;
for_each( v.begin(), v.end(),
        [&sum]( int x ) { sum += x; });
```

Why Language Extensions?

例えば、C言語を使う人の本音は

- マクロだけで頑張るのはホントは辛い
- ポインタを安全に使いたい、頑張ってるけどさ
- エラー処理を確実にしたい、頑張ってるけどさ
- もっと便利な機能を使いたい！

そこでCの拡張です！

Why Language Extensions? part2

例えば、WindowsでC++を使う人の本音は

- ファイルの操作、もっと楽に出来ないのかなぁ
- 表示といえば、MFC/DirectX/GDI…なんでこんなにコーディング量多いの…(￣д￣) ｴｰ
- いつも「それC#がよしなにやってくれるよ」と言われる
- Java/C#使いたくない！！

そこでＣ＋＋の拡張です！

C++/CX

C++言語の拡張

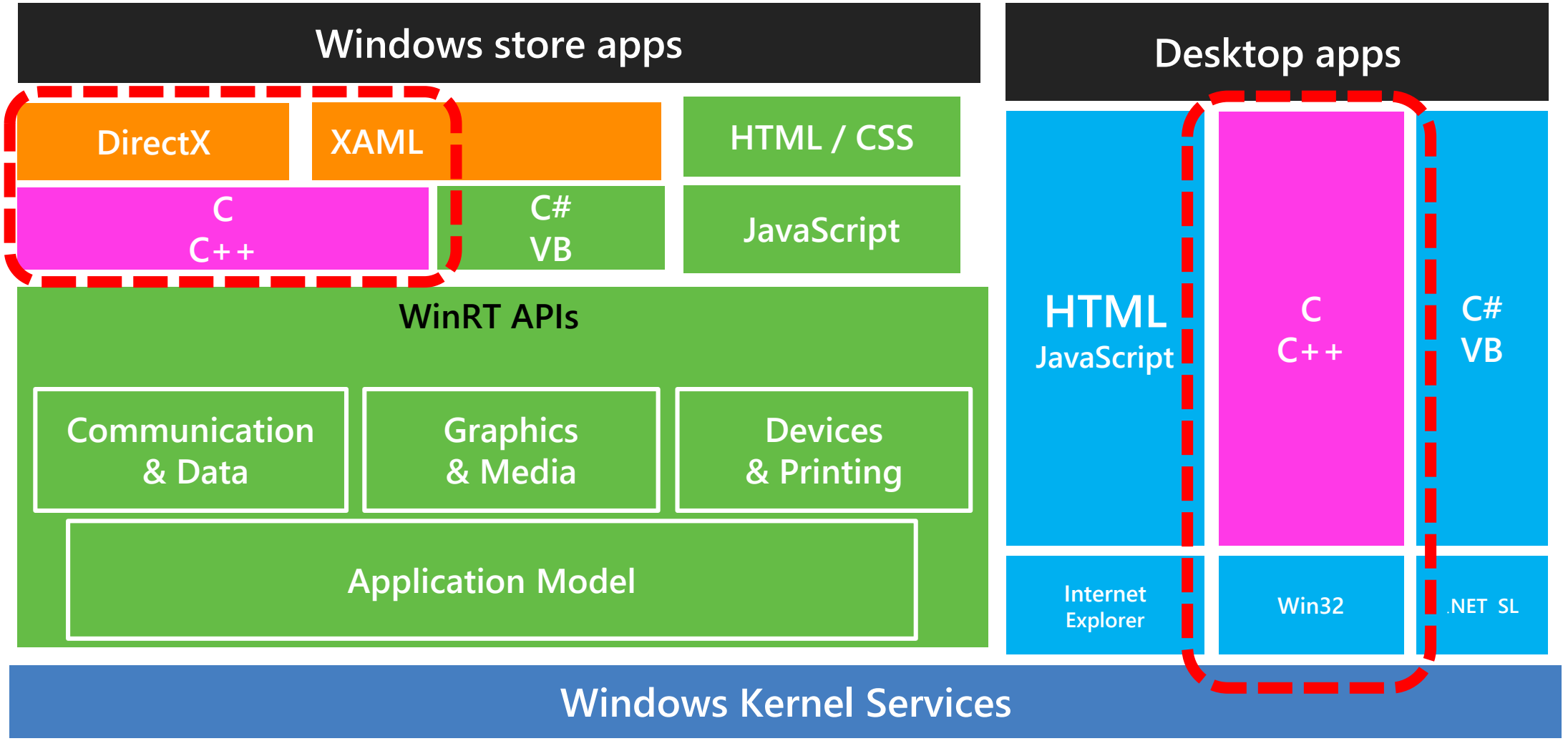
　Visual C++ コンポーネント拡張のことを指す

Windows ストアアプリの作成用

※ストアアプリ以外でも利用可能

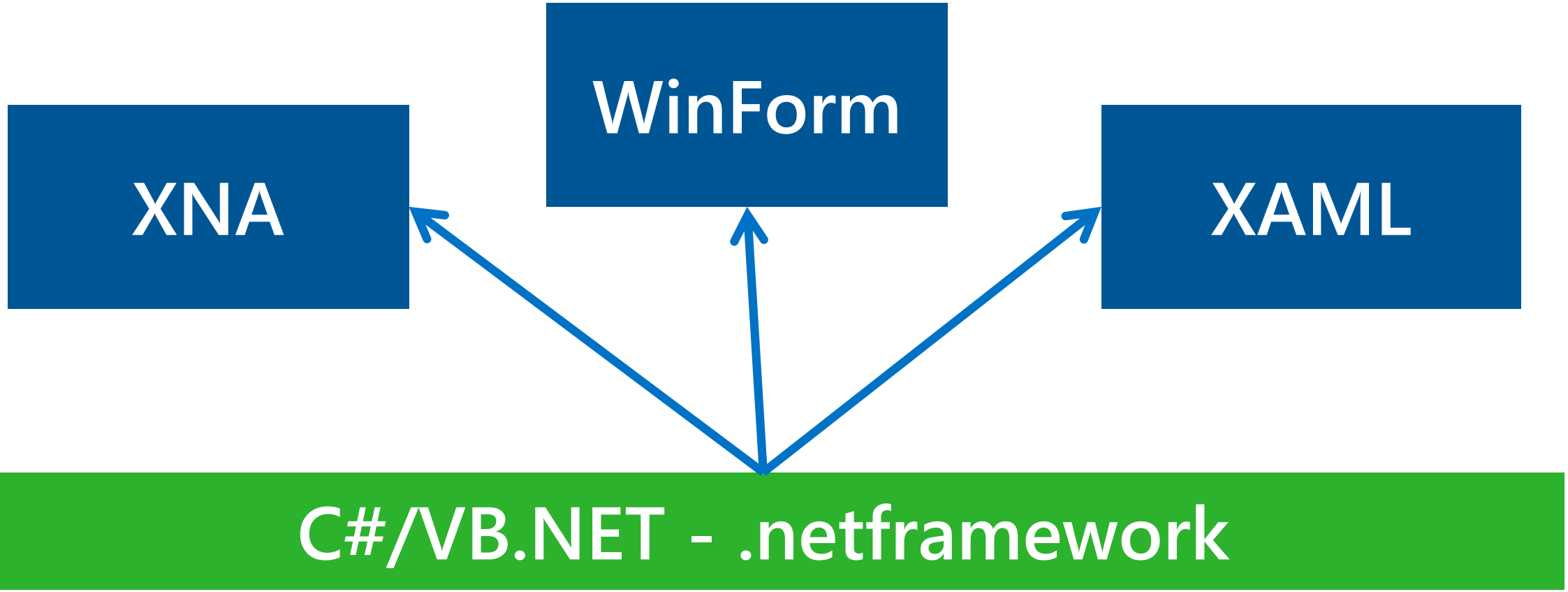
2. Windows application - UI

Programing model



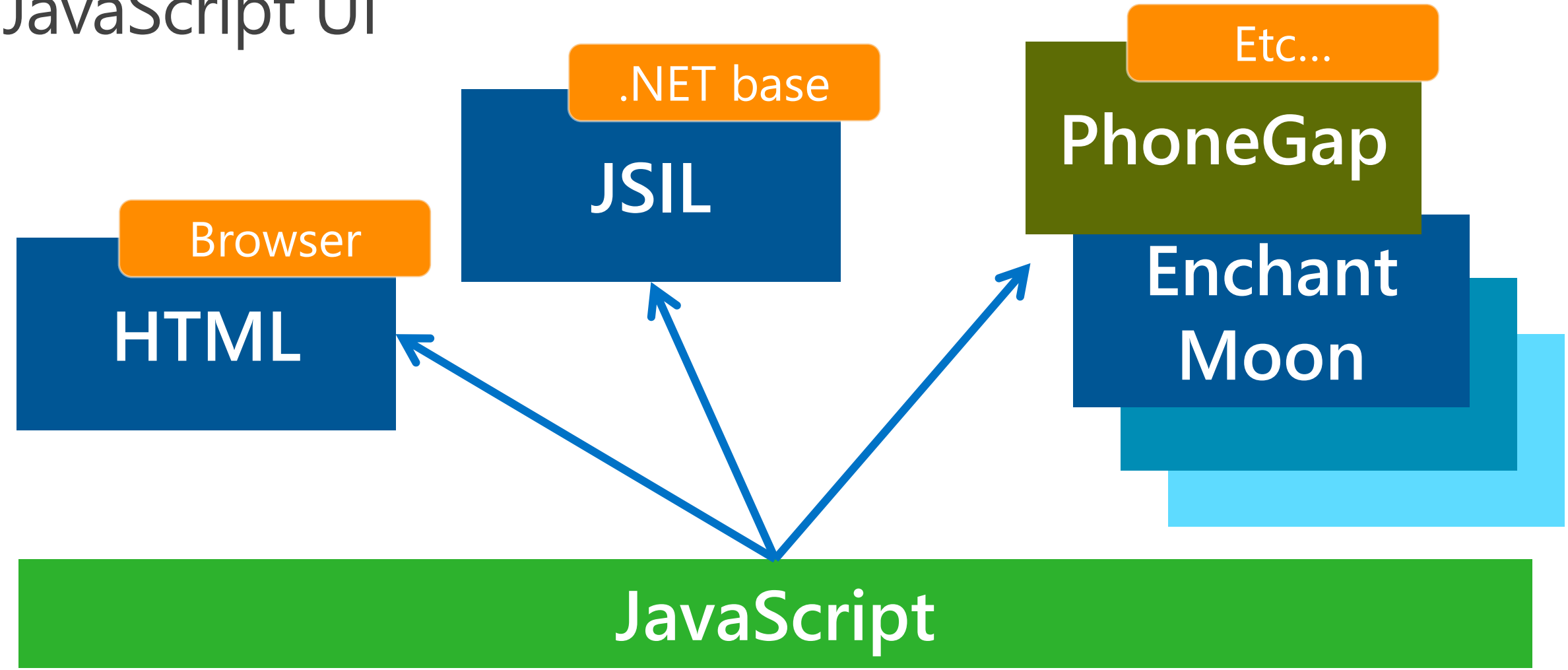
Windows apps - C#/VB.NET

C#/VB.NETでのUIの表現



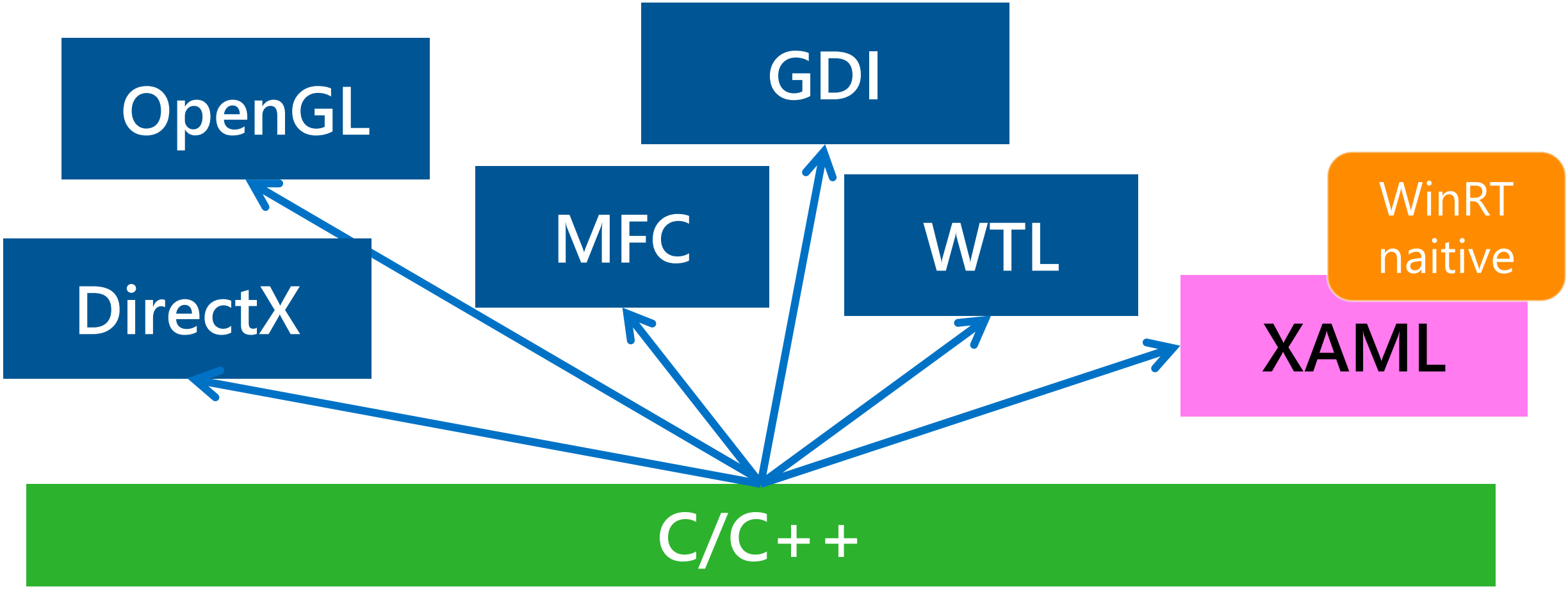
Windows apps - JavaScript

JavaScript UI



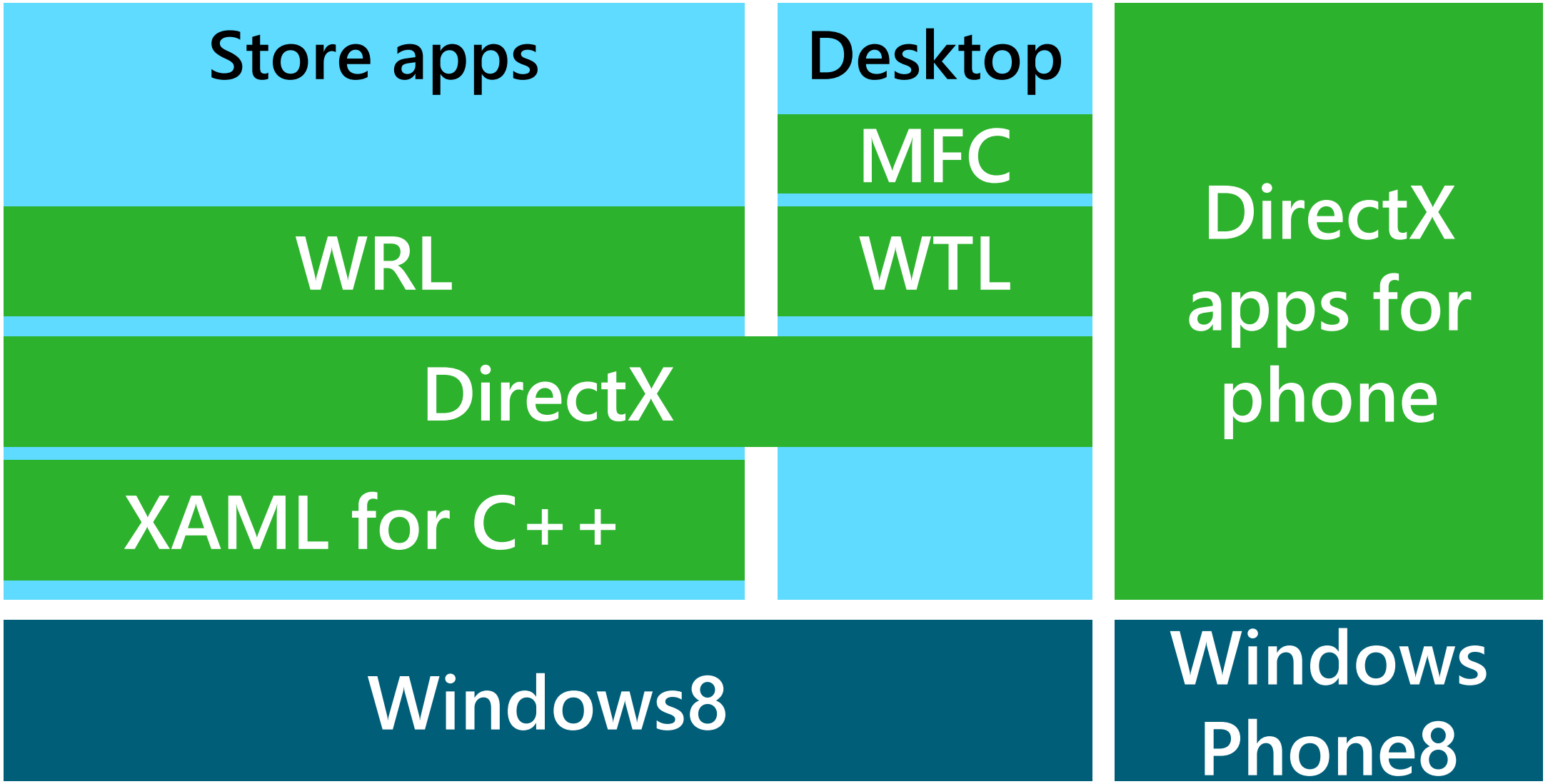
Windows apps - C++

C++から見ると簡単に表現する方法が増えた！



3. How can you use Visual Studio 2012?

Visual Studio 2012のC++の世界



Windows Runtime C++ Template Library (WRL)

Windowsストアアプリ でCOMコンポーネントなど ローレベルアクセスを行うTemplate Library

→ WindowsRuntimeは使わず

ATLで記述されたCOMコンポーネントを簡単に移植、操作可能

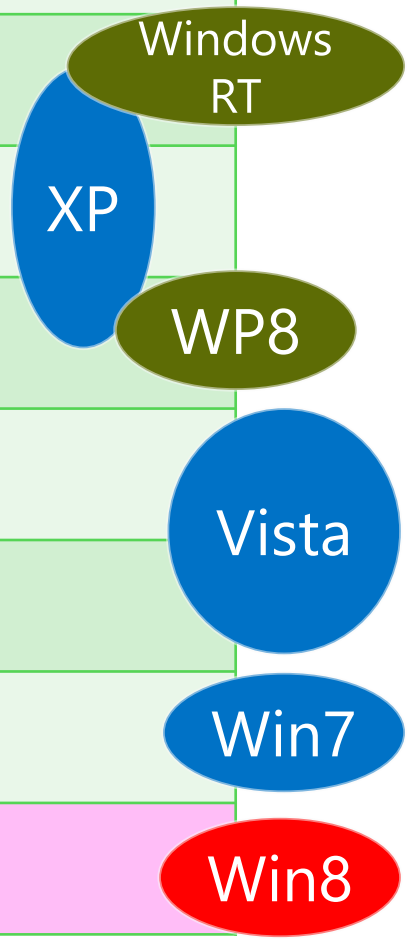
WTL:Windows Template Library – ATL:Active Template LibraryというCOMプログラミングを簡単にするためのテンプレートベースのライブラリ、ダイアログ、フレーム、GDIなどなど,MFCより軽い

DirectX

Windows環境にビデオやサウンドなどのマルチメディア機能を提供するためのAPI

Windowsストアアプリは、DirectX11.1の書き方のみ

機能レベル	ShaderModel
D3D_FEATURE_LEVEL_9_1	2.0
D3D_FEATURE_LEVEL_9_2	2.0
D3D_FEATURE_LEVEL_9_3	2.0
D3D_FEATURE_LEVEL_10_0	4.0
D3D_FEATURE_LEVEL_10_1	4.1
D3D_FEATURE_LEVEL_11_0	5.0
D3D_FEATURE_LEVEL_11_1	5.0



RTならFeature Levelは「**D3D_FEATURE_LEVEL_9_1**」ベース

XAML for C++

C++/CX

C#/VB.NETと同様のUI表現-XAMLが利用可能

XMLベースでUIを表現する

```
<Grid>
  <Button x:Name=“btn00” Content=“押してね“  Click="btn00_Click"/>
  <TextBox x:Name="txt00" Text="何が出るかな？"  FontSize="36"/>
</Grid>
```

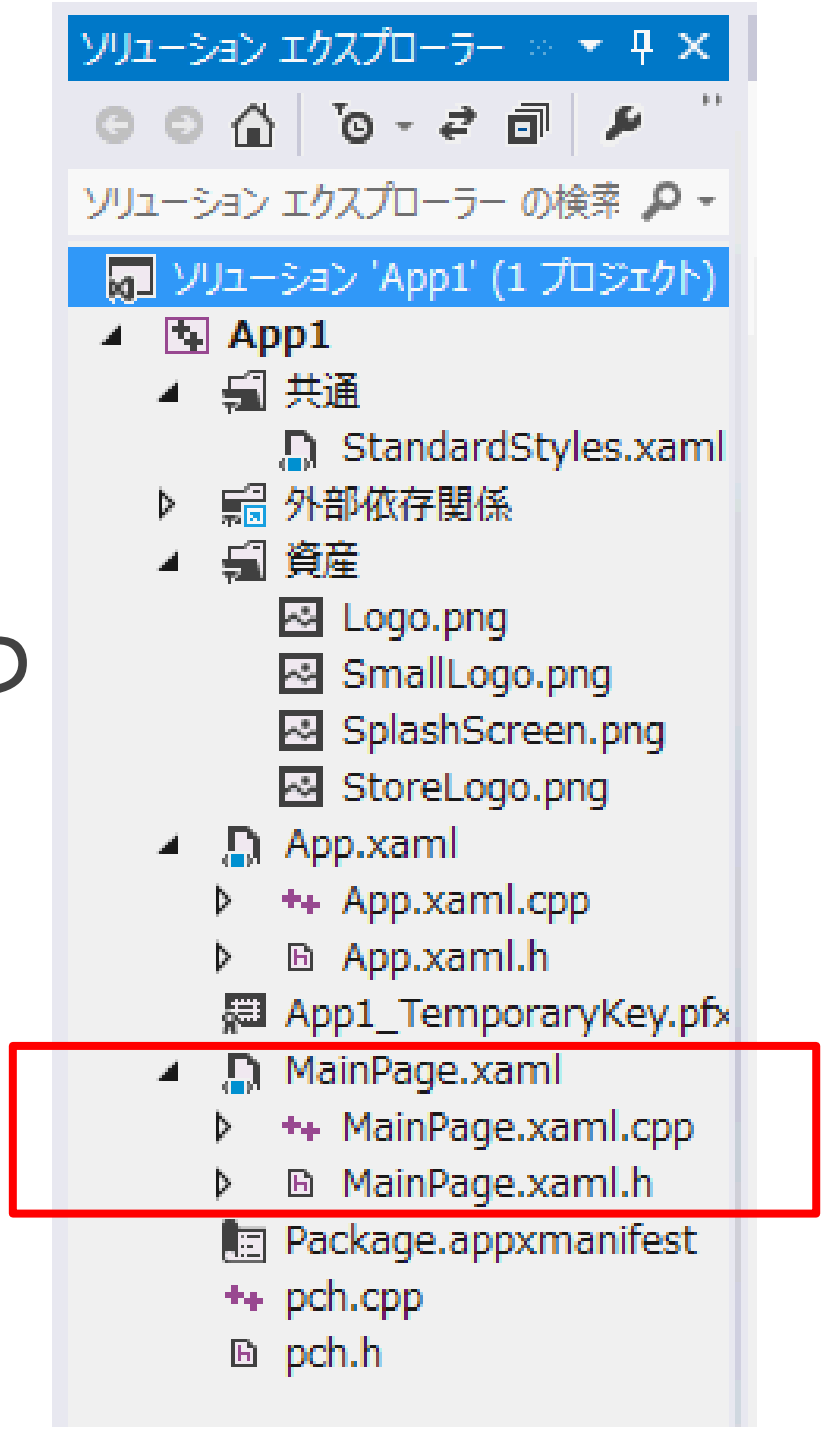
4. Windows store apps – C++/CX

Buildの流れ－C++編

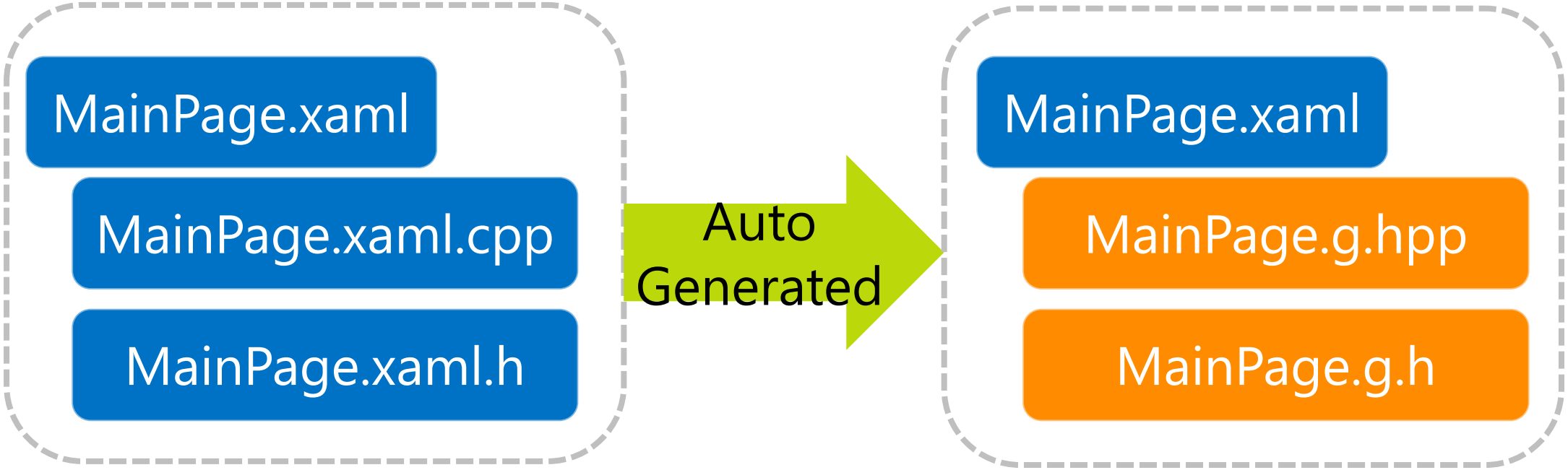
WinRT：WindowsストアAppsの
プロジェクトファイル

なんとなく触れそうなファイルは３つ

MainPage.xaml
MainPage.cpp
MainPage.h



Windows store apps – C++ build



見慣れないソースコードが

しれっと自動生成されます

XAML Objを利用可能なpointerに

```
<Button x:Name="btn00" Content="Cleck me :)"/>
```

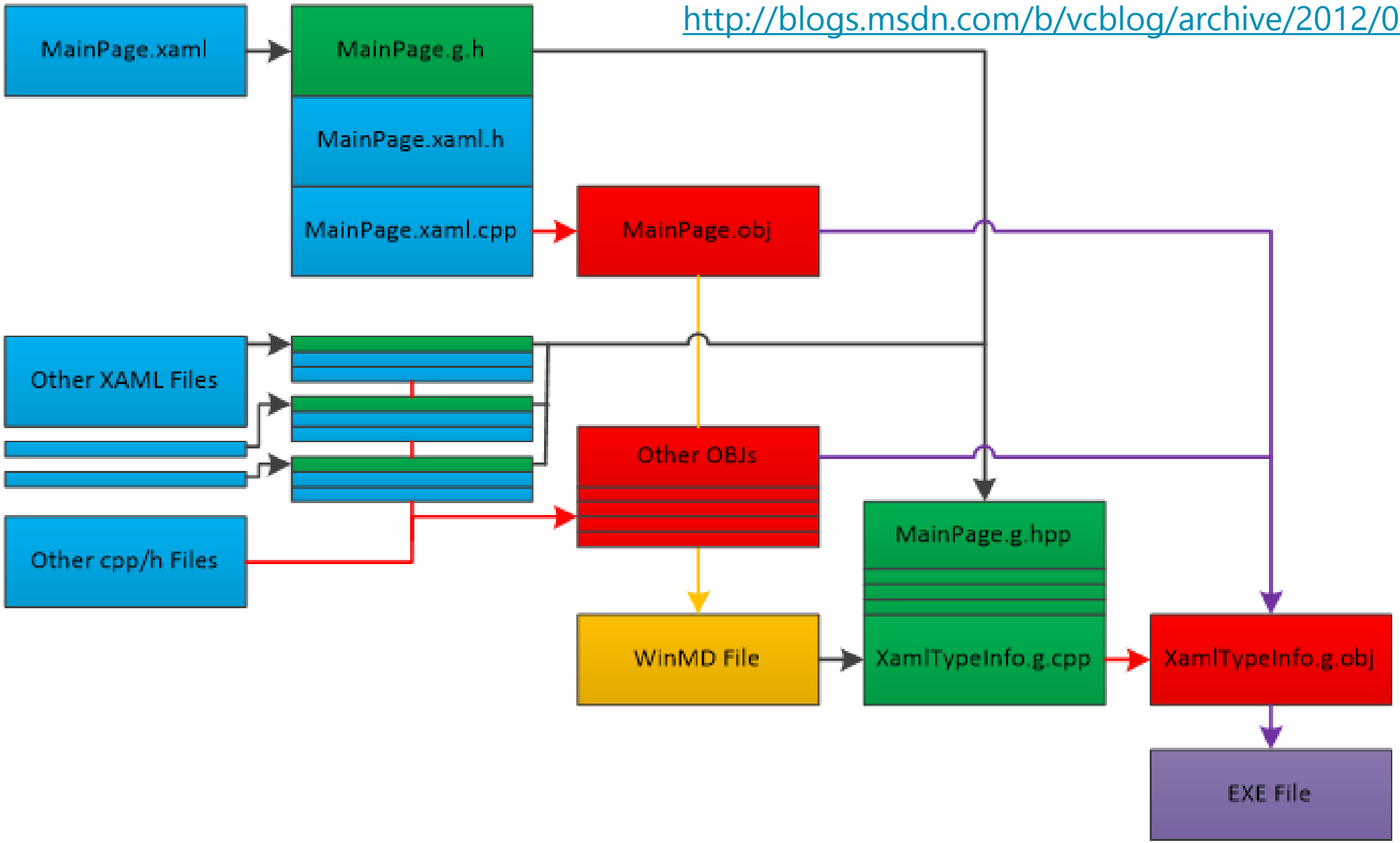
MainPage.xaml

```
void ::StoreApp1::MainPage::InitializeComponent()
{   ...
btn00 = safe_cast<::Windows::UI::Xaml::Controls::Button^>
        (static_cast<Windows::UI::Xaml::IFrameworkElement^>
        (this)->FindName(L"btn00"));
... }
```

MainPage.g.hpp

WinRT C++ Build process

出典：Visula C++ Team Blog - Connecting C++ and XAML
http://blogs.msdn.com/b/vcblog/archive/2012/08/24/10343401.aspx



.winmdファイルのおかげ

WinRTのクラス名やメソッド名などの
メタデータが入っているファイル

.NET, C++, javaScript が共通で理解できる

→*.lib ファイルに似てます！

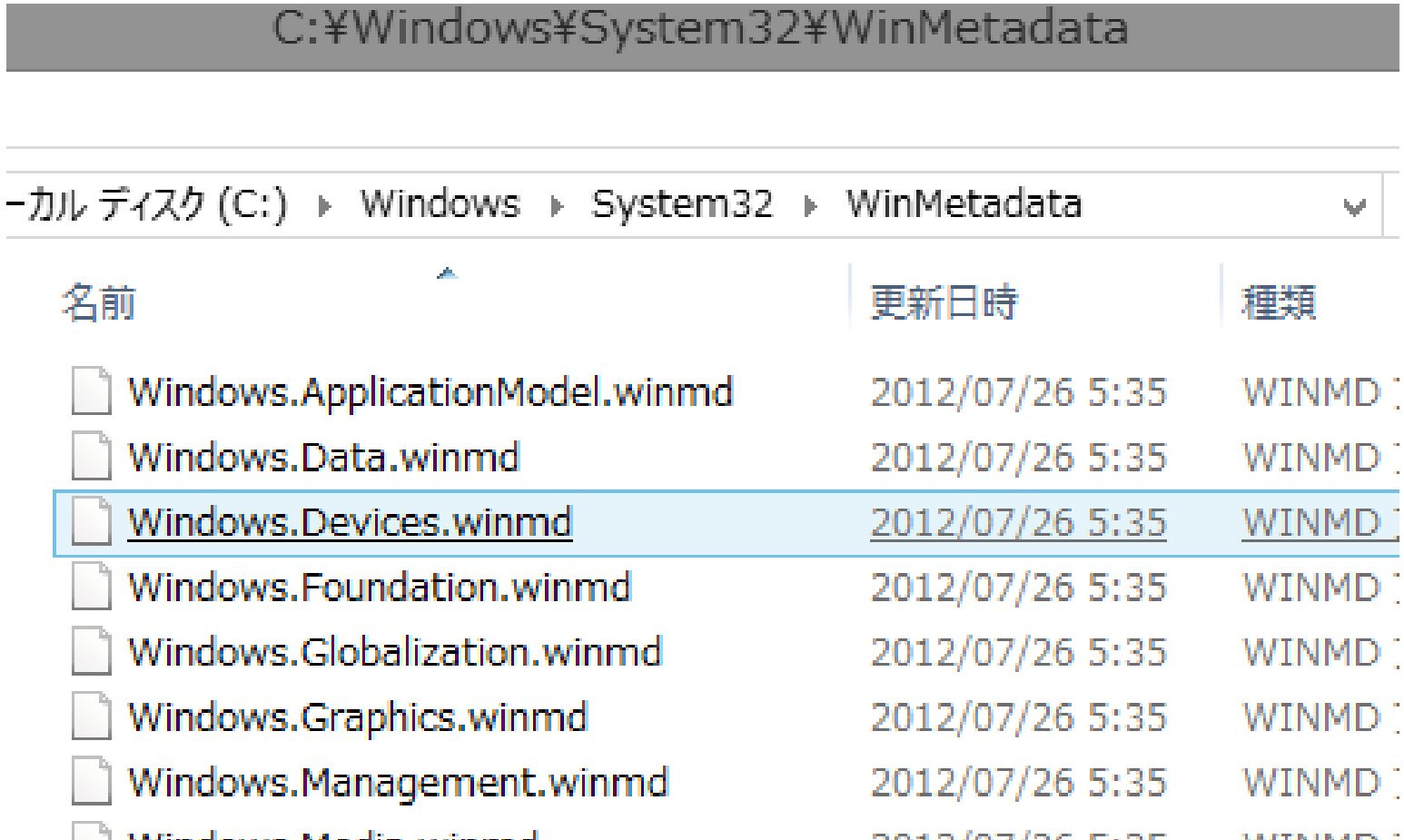
→.NETアセンブリと同じ(binary format)

***.winmdファイルが**
VisualStudioでコンパイルする時に
WinRTのDLLの場所を教えてくれる

Windows meta data

.winmdファイル

C:¥Windows¥System32¥WinMetadataフォルダ



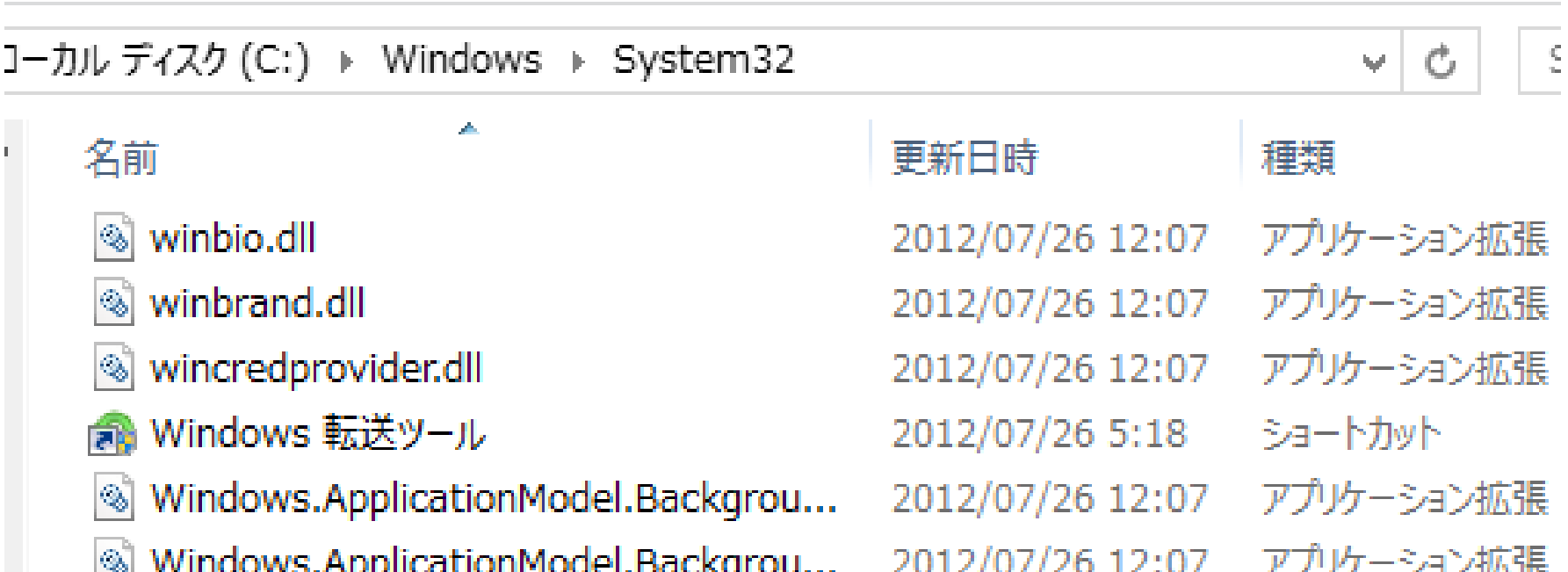
WinRTの実体

C:¥Windows¥System32 の下にあるよ

Windows.〇〇〇.dll

実体は
DLLだったんですね！！

C:¥Windows¥System32



C++/CX が Windows メタデータ (.winmd) ファイルを読み取ると…

良い感じにマップしてくれる例

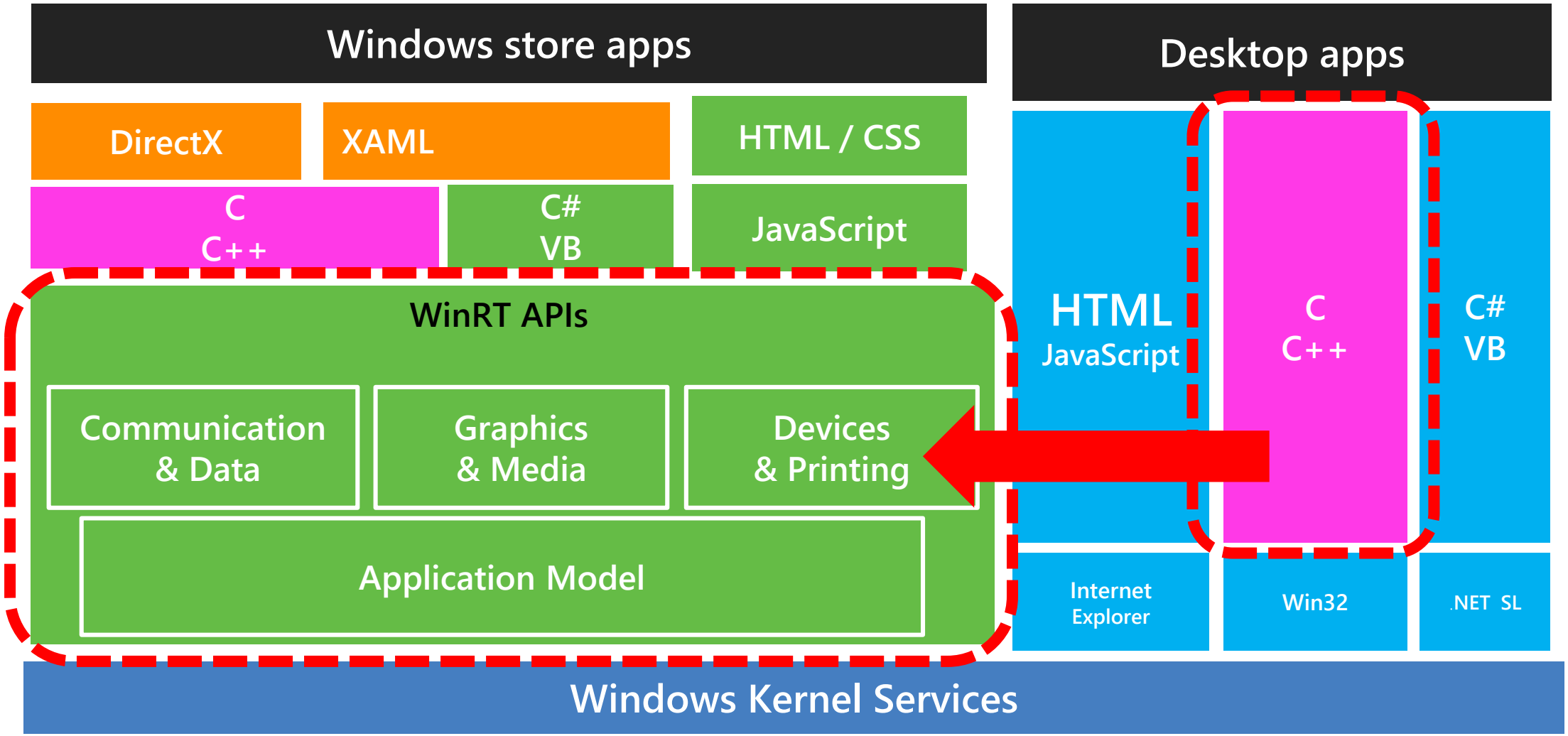
Windows::Foundation の HSTRINGハンドルは…

言語	HSTRINGのマップ
C++/CX	Platform::String class
JavaScript	String object
C#/VB.NET	System.String class

WinRTはAPI

単なるインタフェース 実体はDLL

デスクトップC++からも呼べるよ
※XAML/HTML系以外



WindowsストアアプリのWindowsAPI

- **Windows Runtime**

  C#, VB, C++に提供されているAPI

  UI 関係は Windows.UI.Xaml （★）

- **Windows Library for JavaScript**

  JavaScript API (各種コントロール、CSS)

  （★）に相当するのは WinJS 名前空間

デスクトップアプリからもWindowsRuntime利用可能

Win32コンソールアプリケーション作成

int main(Platform::Array<Platform::String^>^ argv)

コンパイルオプション /ZW つける、/Gm- を外す

.winmdファイルを2つ参照追加

- Windows.winmd - C:¥Program Files (x86)¥Windows Kits¥8.0¥References¥CommonConfiguration¥Natural
- Platform.winmd - C:¥Program Files (x86)¥Microsoft SDKs¥Windows¥8.0¥ExtensionsSDKs¥Microsoft.VCLibs¥11.0¥References¥CommonConfiguration¥natural

5. How to learn C++/CX

C++/CXを学習する

- WinRT APIが利用可能

  →ネイティブライブラリ、XAMLが利用可能

- DirectXの利用可能

XAMLオブジェクトの中がDirectXの世界

- 純粋にネイティブアプリ – WRL利用可能
- XAML からの DirectX利用可能

  →最初はいいと思ったけど後で良いかも

Why do we use C++?

今、C#/VB.net で作っている人

そのまま、C#で書けばいいよ！

C++やDLLを呼び出す部分は完備されてます

|

でも、勘違いしないで！！

Why do we use C++?

今まではC#/VB.NETでしかできなかった

（仕方ない）

XAML系での最終表現の形態がそれしかなかった

でも、今は違います！！

Why do we use C++?

WinRTにおけるC++の土俵

C#が出来ることと全く同じことが可能

C#で出来るからC#を利用する時代ではない

表現出来る土表は全て整っている

問題は…

C++プログラマに

C++/CXつまりWindowsストアアプリを

作るための情報が必要に行きわたっていないこと

正しいC++でのWindowsストアアプリの使い方を伝えていないことにある

（だってXAML系の使い方に慣れてるのはC#/VB.NET系ユーザなんだから、仕方ないでしょ）

ストアアプリでC++を選択したら…

MSDN : C++ を使った Windows ストア アプリのためのロードマップ

http://msdn.microsoft.com/ja-jp/library/windows/apps/xaml/hh700360.aspx

Windowsストアアプリのための C++情報まとめ

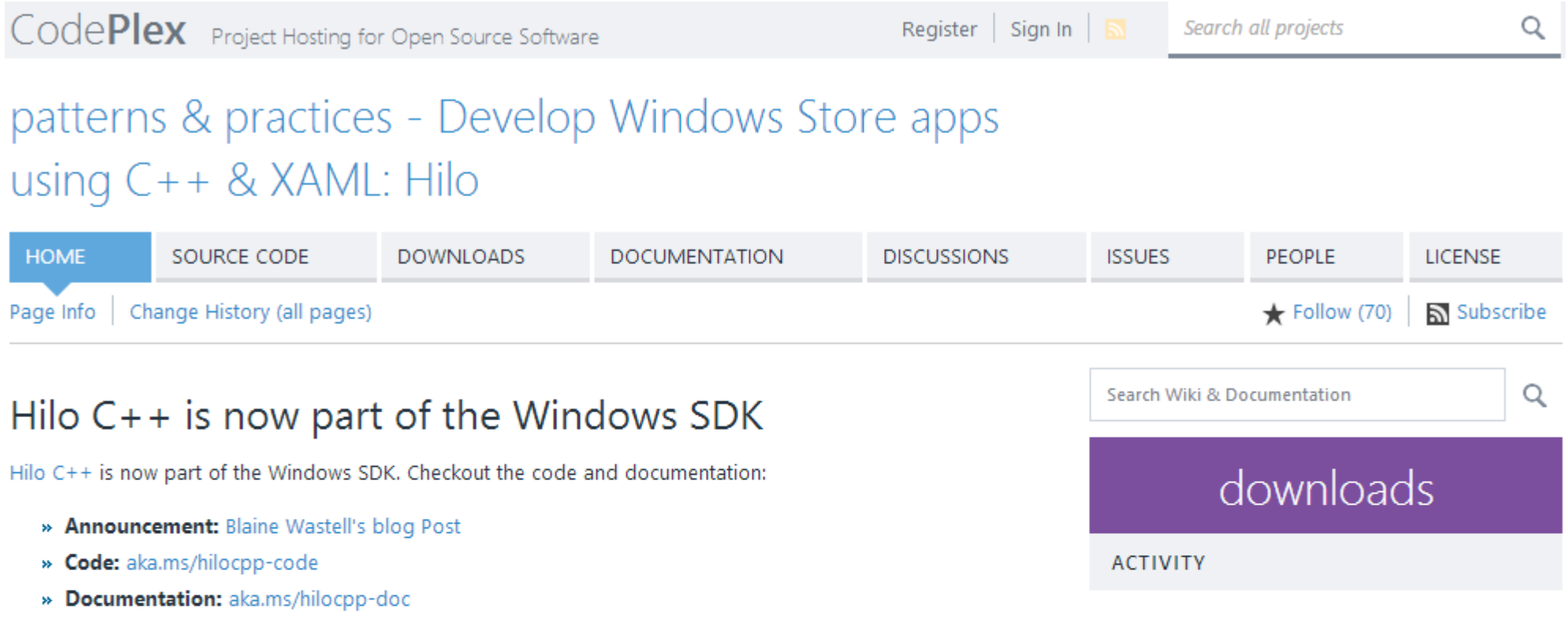
- concurrency::create_async …
- task – then(), when_all() …
- casablanca – C++ REST SDK bindings to HTTP, JSON, and URIs

Hilo

http://hilo.codeplex.com/

Windows7の時はDirectXでした
今はC++/XAML版と、JavaScript版があるよ

C++とXAMLを利用するための実践的サンプル



Hilo利用の前提条件

- Windows 8
- Microsoft Visual Studio 2012
- **C++ と XAML のプログラミングに対する関心**

  →C++と異なり、XAMLはUIを宣言型で書ける

Hiloで学べること

- UX の設計
- 非同期プログラミング パターンとヒント
- タイルとスプラッシュ画面の操作
- Model-View-ViewModel (MVVM) パターン使用
- リポジトリ パターンの使用
- ページの作成とページ間の移動
- コントロールの使用
- タッチの使用
- 中断、再開、アクティブ化の処理
- パフォーマンスの向上
- Windows ストア アプリのテストと展開

巨大なリンク集＆プログラム解説

http://msdn.microsoft.com/ja-jp/library/windows/apps/jj160316.aspx

初心者向け表記

範囲に基づく for ループ

データのコレクションは、範囲に基づく **for** ループを使って処理します。範囲に基づく **for** ループは、反復子やキャプチャ句を使う必要がないため、通常の **for** ループや **std::for_each** アルゴリズムと比べて構文が簡潔です。次にその例を示します。

FileAllPhotosQuery.cpp

C++

```
auto photos = ref new Vector<IPhoto^>();
for (auto file : files)
{
    auto photo = ref new Photo(file, ref new NullPhotoGroup(), policy);
    photos->Append(photo);
}
```

詳しくは、「アルゴリズム (Modern C++)」をご覧ください。

Hilo

C++/CXプログラミングノウハウ

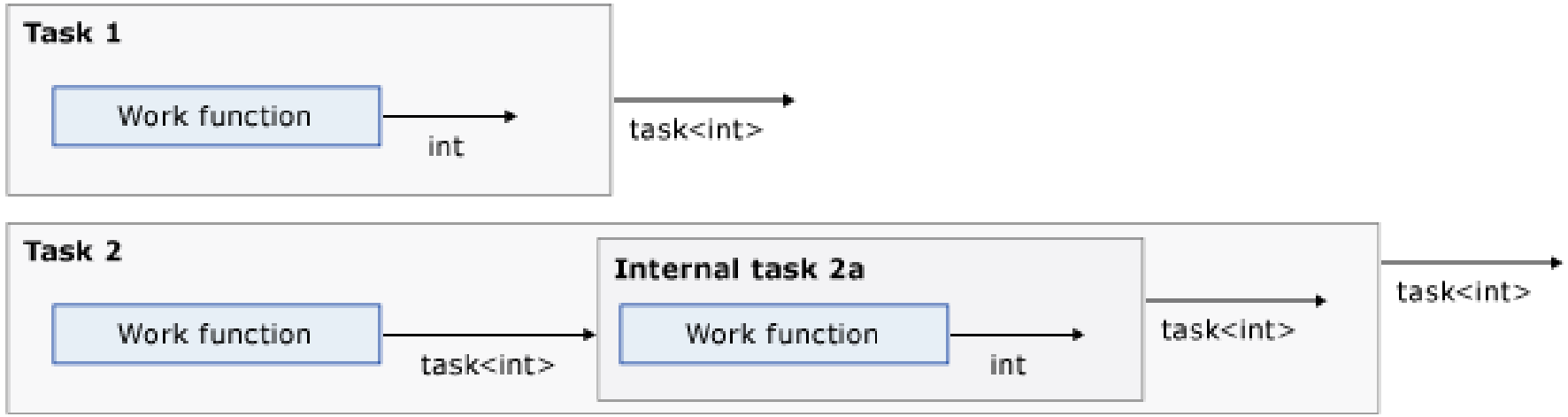
C++/CX を相互運用レイヤーとして使うにあたってのヒント

言語の境界を越えた相互運用性に関して、私たちが Hilo の 作成を通じて得たヒントをここでいくつか紹介します。C++ による Windows ストア アプリについて、言語の境界を越えた相互運用性を 実現する言語拡張機能を包括的に説明したドキュメントが必要な場合は、「Visual C++ 言語のリファレンス (C++/CX)」をご覧ください。

- 型変換のオーバーヘッドに注意する。
- ref クラスのメソッドは決められたスレッドから呼び出す。
- public ref クラスのデストラクターは virtual としてマークする。
- ref クラスは相互運用を目的としてのみ使う。
- マーシャリング コストを最小限にする手法を使う。
- オブジェクト ブラウザーを使ってアプリの .winmd 出力の内容を把握する。
- C++/CX がニーズを満たしていない場合は、WRL を使った、より深いレベルでの相互運用性を検討する。
- C++/CX 言語拡張機能と C++/CLI を混同しない。
- 内部型をパブリック ref クラスで公開しない。

Hilo

taskチェーン – 処理流れ、チェーンの取り消し



図のタスク 1 は、int を返す処理関数を持っています。タスク 1 の型は、task<int> です。タスク 2 は、task<int> を返す処理関数 (図のタスク 2a) を持っています。タスク 2 はタスク 2a が完了するのを待機し、タスク 2a の処理関数の結果をタスク 2 の結果として返します。

タスクのラップ解除は、関連するタスクのネットワークに 条件付きデータ フロー ロジックを適用するのに役立ちます。PPL によるタスクのラップ解除が必要なシナリオと例については、このページの後の方の 「条件付きロジックでの入れ子になった後続タスクの使用」をご覧ください。戻り値の型の表については、 「C++ での非同期 プログラミング (Windows ストア アプリ)」の「ラムダ関数の戻り値の型とタスクの戻り値の型」 をご覧ください。

ImageBrowserViewModel.cpp

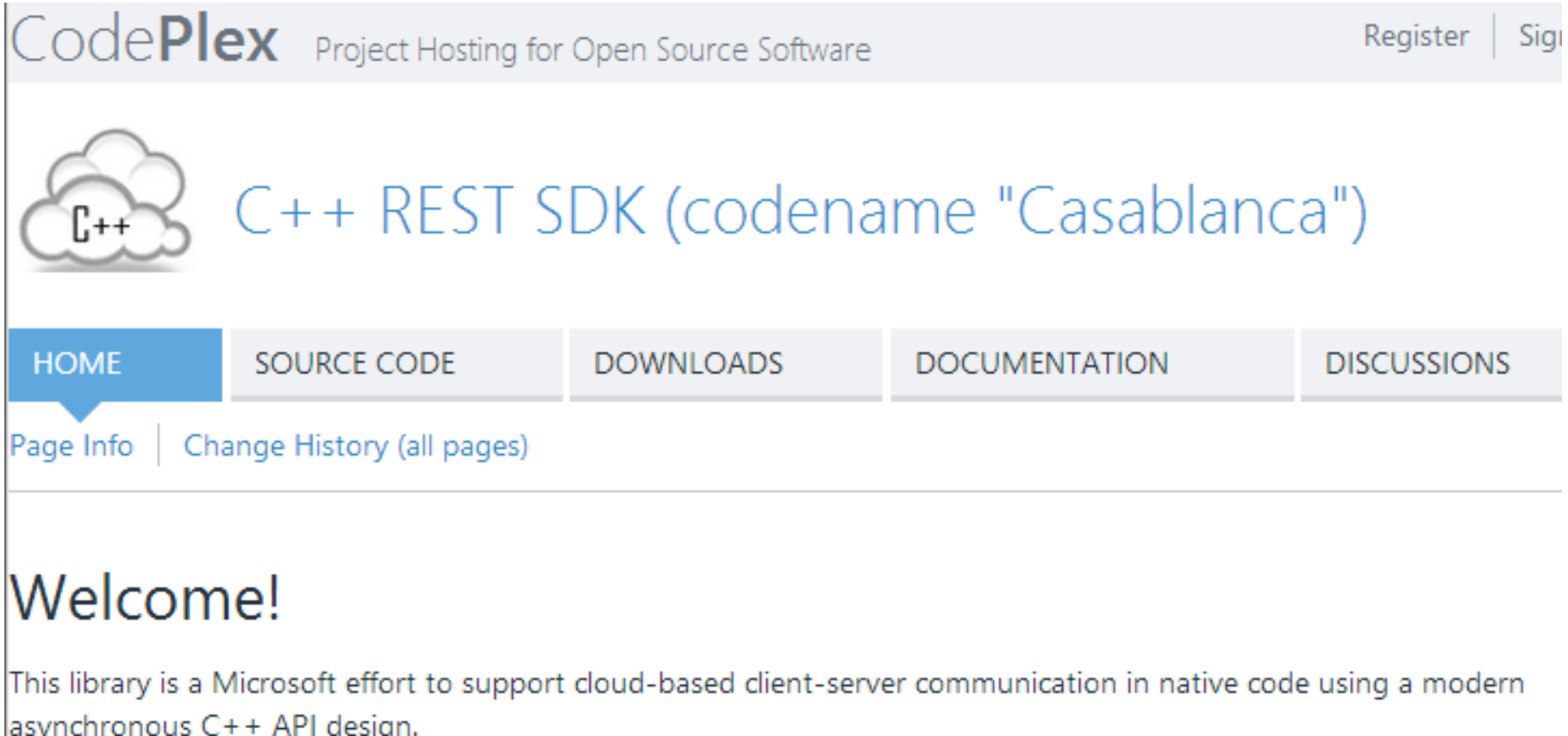
```C++
void ImageBrowserViewModel::StartMonthAndYearQueries()
{
    assert(IsMainThread());
    assert(!m_runningMonthQuery);
    assert(!m_runningYearQuery);
    assert(m_currentMode == Mode::Active || m_currentMode == Mode::Running || m_currentMode

    m_cancellationTokenSource = cancellation_token_source();
    auto token = m_cancellationTokenSource.get_token();
    StartMonthQuery(m_currentQueryId, token);
    StartYearQuery(m_currentQueryId, token);
}
```

StartMonthAndYearQueries メソッドは、**concurrency::cancellation_token_source** と **concurrency::cancellation_token** オブジェクトを作成し、**StartMonthQuery** と **StartYearQuery** メソッドにキャンセル トークンを渡します。**StartMonthQuery** メソッドの実装は、次のとおりです。

Casablanca – C++ REST SDK

HiloでWindowsストアアプリの作り方を覚えたら、
次は通信がしたくなる
http://casablanca.codeplex.com/



Casablancaとは

- RESTサービス(Windows Vista/7/8, Linux)
  VS2010/2012, クラウドベースの通信サポート
- 非同期処理、ストリームバッファ
- HTTP Client
- C++11 base
- Apache オープンソース

Azureなど

-. Welcome back to Windows Phone

Using C++ in your Windows Phoen Apps

Windows Phone8のC++サポート

- VC++ 2012
- Windows8と同じコードでOK
- C++11 ベース
- DirectX (XAML は利用できない)

summary

まとめ

- クライアントサイドアプリケーション復権
  C++ネイティブコード
- C++11規格
  拡張C++/CX
- XAMLでの表現力
- Windows RuntimeはAPI
- XAML × C++ の学習 Hilo / Casablanca

これからも

C++でアプリケーションを作りましょう

そのための道は

マイクロソフトが全て用意してくれています