ドキュメントとエラーハンドリング

高橋 晶(Akira Takahashi)
id:faith_and_brave
@cpp_akira

エラーハンドリング勉強会 2011/09/04(日)

自己紹介

- 株式会社ロングゲート 取締役
- C++テンプレートメタプログラマ
- Boost Geometry Libraryコントリビュータ
- 『プログラミングの魔導書』編集長。

チュートリアル

- チュートリアルにはエラーハンドリングが記載されていないことが多い。

- そのまま使用すると、あとで想定していなかったエラーが発生してしまう場合がある。

ドキュメントは大事！

- 適切なエラーハンドリングを行うためには、ドキュメントをしっかり読まないといけない。

- その関数はどういったケースでどういったエラーを出力する可能性があるのか。
  そのエラーは、そのプロジェクトで起こることが想定できるものか。

ドキュメントの読み方

- 直接記載されていないエラーが報告される可能性がある。
- その関数自体だけではなく、引数として渡したオブジェクトがエラーを報告する場合もある。
- その関数が引数をどのように扱うのか注意して読む必要がある。

アンドキュメントなものは使わない

- アンドキュメントな機能は、将来のバージョンアップで予告なく変更・削除される可能性がある。
- 「ライブラリをアップデートしたらリリースノートにも変更が書かれていないのに動かなくなった」では困る。
- アンドキュメントだが有用な機能だと思うなら、それを作者に報告し、方針を確認した上で使おう。

  例.
  「アンドキュメントだからまだ使わないほうがいい？」
  「ドキュメントパッチを送れば取り込んでくれる？」

ドキュメントに記載されていないエラー

- ドキュメントに記載されていないエラーを投げる可能性が、実装を読んで発覚する場合がある。
- ドキュメントバグなので報告しよう。
- アンドキュメントなエラーをハンドリングする場合は、その旨コメントに書こう。
  報告した際のチケット番号を記載するとGood!

ドキュメント読むのめんどくさい

- めんどくさいです。
- ドキュメント読まなくても適切なエラーハンドリングがしたい！
- ライブラリ作者は、間違った使い方をした場合や、適切なエラーハンドリングを行わなかった場合に、事前にそれを検知できる機構を用意すべし。

型付けしよう

- 静的型付け言語では、型付けによって、間違った使い方をした場合にコンパイルエラーにすることができる。
- ハンドリング洩れも、がんばればチェックできると思う(戻り値を受け取ったかのチェック、エラーチェックしたか、をコンパイル時に判断するのは難しい)。
- ただし例外の型付け、ハンドリング洩れの静的チェックは難しい。IDEのサポートに期待。
(もしくは例外を積極的に使わない)

まとめ

- ドキュメントは大事
- アプリケーションコードの全ての関数までドキュメントを記載する必要はないが、ライブラリコードはドキュメントを書くべし。
- ドキュメントが間違っていたら報告しよう。
  それはあとに続く開発者にとって有益。
- 型は大事。
  型はプログラム内にプログラムで記載されたドキュメント。