エクセルソフト Web セミナー
「インテル® コンパイラー入門（Linux* 版）」

[対象製品]
・インテル® C++ コンパイラー 9.1 Linux 版
・インテル® Fortran コンパイラー 9.1 Linux 版

XLSOFT

本セミナーの内容

1. インテル® コンパイラーの特長

2. 開発環境

3. インストール手順

4. コンパイル

5. 最適化オプション

インテル® コンパイラーの特長

その1　GCC との互換性

- GCC 3.2/3.3/3.4 とのソースおよびオブジェクト・レベルでの互換性があり、既存のアプリケーションを再コンパイルして、パフォーマンスを向上することができます。

その2　高度な最適化機能

- インテル® コンパイラーでコンパイルするだけで高速なアプリケーションを作成できますが、最適化オプションを使用することにより、更なるパフォーマンスの向上を期待できます。

その3　Eclipse* IDE との統合

- インテル® C/C++ コンパイラーは、Eclipse と統合してビルド、デバッグ作業が可能です。

その4　充実した並列化機能

- POSIX Thread、OpenMP* へのサポートはもとより、インテル独自の並列化オプションでアプリケーションの高速化に貢献します。

コンポーネント一覧

◆ インテル® コンパイラーには、以下のコンポーネントが含まれます。

コンポーネント	内　容
コンパイラー	・IA-32 / インテル® 64 / IA-64 用コンパイラー
デバッガー	・IA-32 / インテル® 64 / IA-64 用デバッガー
Eclipse 開発環境	・Eclipse SDK 3.1.1 ・CDT 3.0.1 ・JRE 1.5.0_04 ・インテル® コンパイラー統合プラグイン ・インテル・デバッガー統合プラグイン

※ Eclipse 開発環境は、インテル® C++ コンパイラーのみです。

プラットフォーム別コンポーネント一覧

◆ プラットフォームによってインストールできるツールが異なります。

ツール ＼ プラットフォーム	IA-32	インテル® 64	IA-64
IA-32 用 コンパイラー	○	○	
インテル® 64 用 コンパイラー		○	
IA-64 用 コンパイラー			○
IA-32 用 デバッガー	○	○	
インテル® 64 用 デバッガー		○	
IA-64 用 デバッガー			○
※ Eclipse 開発環境	○		○

※ Eclipse 開発環境は、インテル® C++ コンパイラーのみです。

本セミナーの内容

1. インテル® コンパイラーの特長

2. 開発環境

3. インストール手順

4. コンパイル

5. 最適化オプション

開発環境

◆ 以下は、インテル® コンパイラーが正式にサポートする開発環境です。

	IA-32	インテル® 64	IA-64
OS	• Red Hat* Linux 7.3, 8, 9 • Red Hat Enterprise Linux* 2.1, 3, 4 • SUSE* LINUX • Fedora* Core 4	• Red Hat Enterprise Linux 3, 4 • SUSE* LINUX • Fedora Core 4	• Red Hat Linux 7.2 • Red Hat Enterprise Linux AS 2.1, 3, 4 • SGI* ProPack* for Linux 5 • SUSE LINUX
kernel	2.4.x / 2.6.x		
glibc	2.2.4, 2.2.5, 2.2.93, 2.3.2, 2.3.3, 2.3.4, 2.3.5	2.2.93, 2.3.2, 2.3.3, 2.3.4, 2.3.5	2.2.4, 2.2.93, 2.3.2, 2.3.3, 2.3.4, 2.4
その他	• gcc、g++ などの基本開発ツール	• gcc、g++ などの基本開発ツール • 32 ビット用 C/C++ ランタイム・ライブラリー	• gcc、g++ などの基本開発ツール

本セミナーの内容

1. インテル® コンパイラーの特長

2. 開発環境

3. インストール手順

4. コンパイル

5. 最適化オプション

インストール手順

ライセンスファイルの取得

インテル 社 HP からシリアル番号を登録してライセンスファイルを取得する

https://registrationcenter.intel.com/regcenter/register.aspx

インテル® コンパイラーのインストール

（CD-ROM から）

CD-ROM 内の インストール・シェル(./install.sh)を実行

（ダウンロード パッケージ から）

ダウンロード・パッケージ（.tar.gz）を解凍して ./install.sh を実行

インストール・ディレクトリー 構成

◆ 以下の表は、インテル® コンパイラーをデフォルト・インストールした場合のディレクトリー構成図です。

/opt/intel	インテル開発ツール・ルート・ディレクトリー
/cc	インテル® C/C++ コンパイラー（IA-32/IA-64）
/cce	インテル® C/C++ コンパイラー（インテル® 64）
/fc	インテル® Fortran コンパイラー（IA-32/IA-64）
/fce	インテル® Fortran コンパイラー（インテル® 64）
/idb	インテル・デバッガー（IA-32/IA-64）
/idbe	インテル・デバッガー（インテル 64）
/eclipsepackage	Eclipse 開発環境
/licenses	ライセンスファイル

インストール・ディレクトリー 構成（つづき）

- それぞれのコンパイラー・ディレクトリーに含まれるサブディレクトリーは以下の通りです。

/opt/intel/cc/9.1.xxx	インテル® C/C++ コンパイラー（IA-32/IA-64）
/opt/intel/cce/9.1.xxx	インテル® C/C++ コンパイラー（インテル® 64）
/opt/intel/fc/9.1.xxx	インテル® Fortran コンパイラー（IA-32/IA-64）
/opt/intel/fce/9.1.xxx	インテル® Fortran コンパイラー（インテル® 64）
/bin	コンパイラー本体、その他ツール群
/include	インクルード、モジュールファイル
/lib	静的 / 共有 ライブラリー
/doc	ドキュメント一式
/samples	サンプルプログラム（int_sin）

本セミナーの内容

1. インテル® コンパイラーの特長

2. 開発環境

3. インストール手順

4. コンパイル

5. 最適化オプション

コンパイル

インテル® コンパイラーによるコンパイル方法

① コマンドラインからのコンパイル

(C/C++)

icc [*オプション*] file1.c file2.c … ⟵ C の場合

icpc [*オプション*] file1.cpp file2.cpp … ⟵ C++ の場合

(Fortran)

ifort [*オプション*] file1.f90 file2.f90 …

② Make ファイルによるコンパイル

make -f mymakefile

③ Eclipse 開発環境からのコンパイル（※ C/C++ のみ）

iccec コマンドにてEclipse起動

コンパイル（コマンドライン編）

インテル® C++ コンパイラーによるコンパイル手順

1. 環境変数の設定

   ※ PATH, LD_LIBRARY_PATH, INTEL_LICENSE_FILE の設定

   # source /opt/intel/cc/9.1.xxx/bin/iccvars.sh (bash の場合)

   # source /opt/intel/cc/9.1.xxx/bin/iccvars.csh (csh の場合)

2. コンパイル

   # cd /opt/intel/cc/9.1.xxx/samples ←サンプルディレクトリーまで移動

   # icc int_sin.c ← コンパイル

3. 実行

   # ./a.out

コンパイル（コマンドライン編）

インテル® Fortran コンパイラーによるコンパイル手順

1. 環境変数の設定

   ※ PATH, LD_LIBRARY_PATH, INTEL_LICENSE_FILE の設定

   # source /opt/intel/fc/9.1.xxx/bin/ifortvars.sh (bash の場合)

   # source /opt/intel/fc/9.1.xxx/bin/ifortvars.csh (csh の場合)

2. コンパイル

   # cd /opt/intel/fc/9.1.xxx/samples ←サンプルディレクトリーまで移動

   # ifort int_sin.f90 ← コンパイル

3. 実行

   # ./a.out

入力ファイル

インテル® コンパイラーで認識される拡張子

ファイル名	意味	動作
file.c	C ソースファイル	コンパイラーで処理される
file.cpp	C++ ソースファイル	コンパイラーで処理される
file.for file.f	Fortran 固定形式 フォーマット	コンパイラーで処理される
file.f90	Fortran 自由形式 フォーマット	コンパイラーで処理される
file.s	アセンブリー・ファイル	アセンブラーで処理される
file.o	オブジェクト・ファイル	リンカーで処理される
file.a	静的ライブラリー	リンカーで処理される

コンパイルオプション

- コンパイル作業に必要な基本コンパイルオプション

オプション名	意味	例
-o	出力ファイル名の指定	# icc -o output file1.c file2.c # ifort -o output file.f90 file2.f90
-c	オブジェクト・ファイルの作成	# icc –c file1.c file2.c # ifort –c file1.f90 file2.f90
-I（i の大文字）	モジュール、インクルードファイル検索パスの追加 （※カレント・ディレクトリーはデフォルト検索パス）	# icc –I/tmp/myinc file.c # ifort –I/tmp/myinc file.90
-L	ライブラリー・ファイル検索パスの追加	# icc file.c –L/tmp/lib –lmylib # ifort file.f90 –L/tmp/lib –lmylib (libmylib.so → -lmylib)
-l（L の小文字）	ライブラリー・ファイルの指定	

コンパイル 例

- **複数ファイルのコンパイル**
  # icc -o output file1.c file2.c file3.c (出力ファイル “output” が作成される)
- **インクルード・ファイル検索パスの指定**
  # icc -I./myinc file1.c file2.c (./myinc ディレクトリーをインクルード・パスに追加)
- **オブジェクト・ファイルの作成**
  # icc -c file1.c file2.c file3.c (“file1.o”、“file2.o”、“file3.o” が作成される)
- **ライブラリーの指定（静的リンク）**
  # icc file1.c file2.c ./lib/mylib.a (mylib.a が静的リンクされ、“a.out” が作成される)
- **ライブラリーの指定（動的リンク）**
  # icc file1.c -L./lib/test -lmylib (libmylib.so が動的リンクされ、“a.out” が作成される)

デフォルト設定ファイル

(C/C++)

- /opt/intel/cc/9.1.xxx/bin/icc.cfg(icpc.cfg)
- /opt/intel/cce/9.1.xxx/bin/icc.cfg(icpc.cfg)

(Fortran)

- /opt/intel/fc/9.1.xxx/bin/ifort.cfg
- /opt/intel/fce/9.1.xxx/bin/ifort.cfg

これらのファイルに記述されたコンパイルオプションは、コンパイル時に自動的に指定されます。
（※ コンパイル時に指定する手間が省ける！）

ファイル実行

実行ファイルに依存する共有ライブラリー（.so）の確認

- ldd コマンドでライブラリーの依存関係を確認

```
[root@xlsoft samples]# ldd a.out
        libimf.so => not found          <-- 未解決ライブラリー
        libm.so.6 => /lib/tls/libm.so.6 (0x00842000)
        libgcc_s.so.1 => /lib/libgcc_s.so.1 (0x00c9d000)
        libirc.so => not found          <-- 未解決ライブラリー
        libc.so.6 => /lib/tls/libc.so.6 (0x00715000)
        libdl.so.2 => /lib/libdl.so.2 (0x00867000)
        /lib/ld-linux.so.2 (0x006f7000)
```

- 環境変数（LD_LIBRARY_PATH）にパスを追加

```
# export LD_LIBRARY_PATH=/opt/intel/cc/9.1.044/lib:${LD_LIBRARY_PATH}
```

コンパイル（Eclipse編）

[ステップ1] Eclipse の起動

[ステップ2] ワークスペースの指定

[ステップ3] 新規プロジェクトの作成

[ステップ4] プロジェクトにファイルを追加

[ステップ5] プロジェクトのビルド

[ステップ6] 実行構成の作成

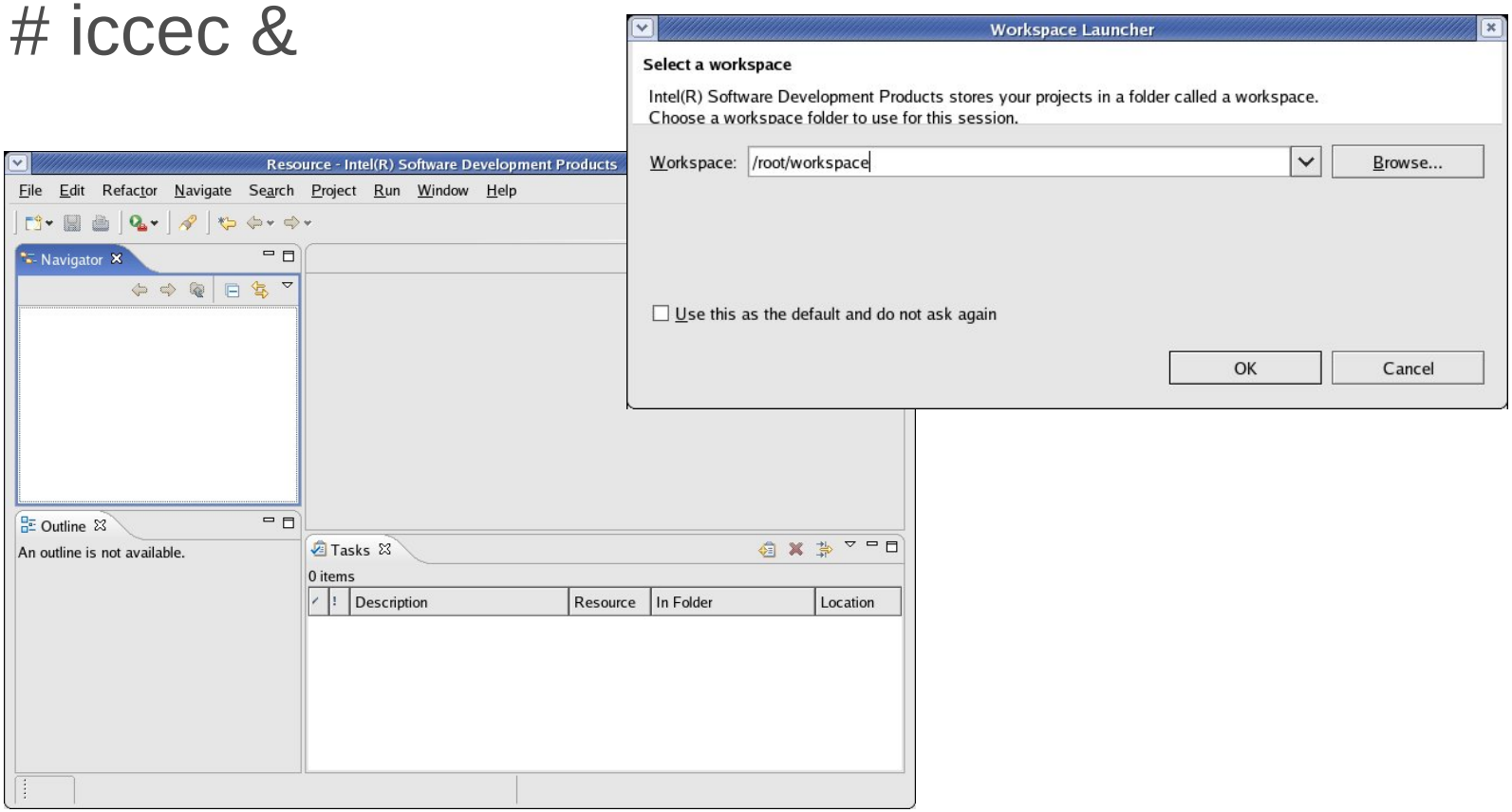
[ステップ7] プロジェクトの実行

Eclipse の起動

- コマンドラインから

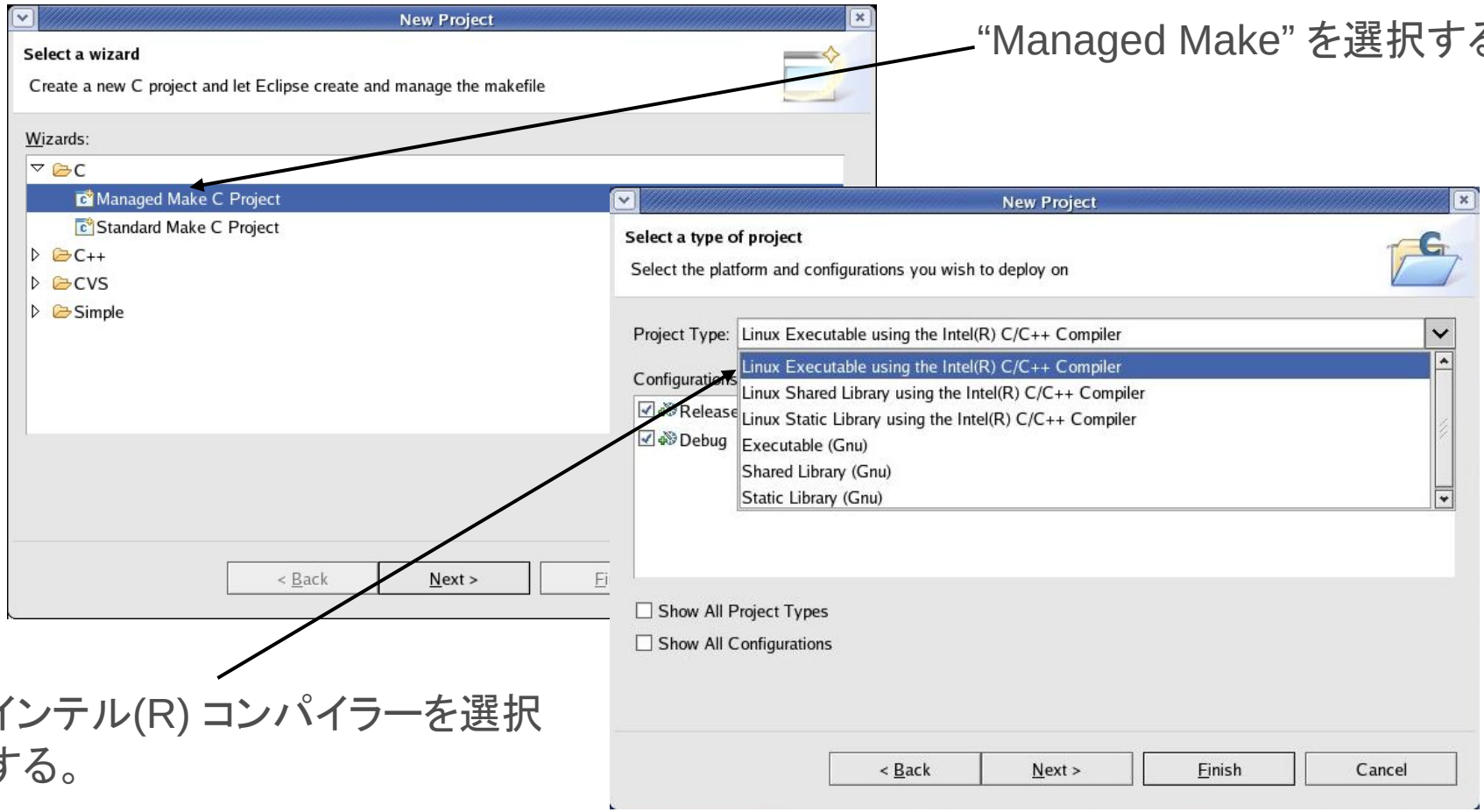
  # source /opt/intel/cc/9.1.xxx/bin/iccvars.sh(csh)

  # iccec &



新規プロジェクト

- [File] – [New] – [Project…] から新規プロジェクトを作成

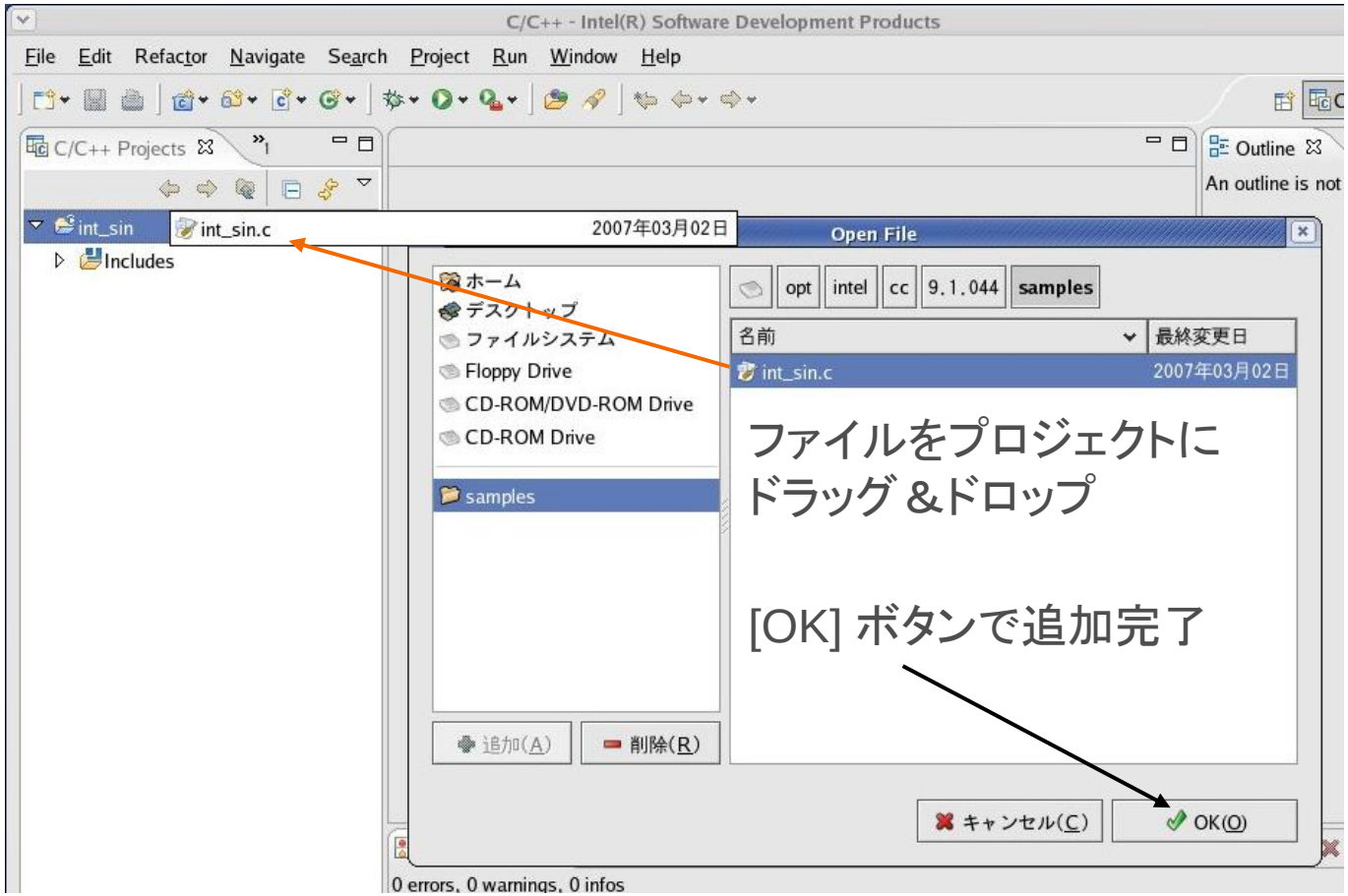


“Managed Make” を選択する。

インテル(R) コンパイラーを選択する。

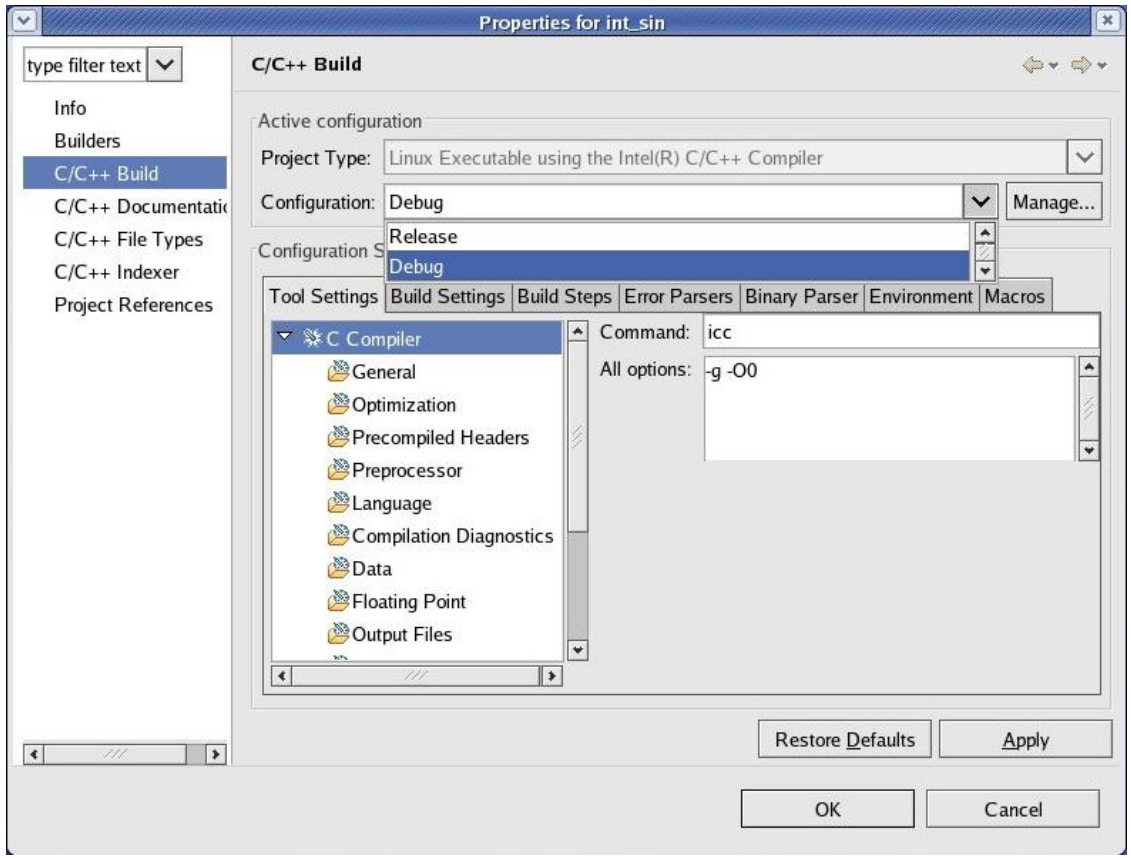
ファイルの追加

- [File] – [Open File…] からファイルをプロジェクトに追加



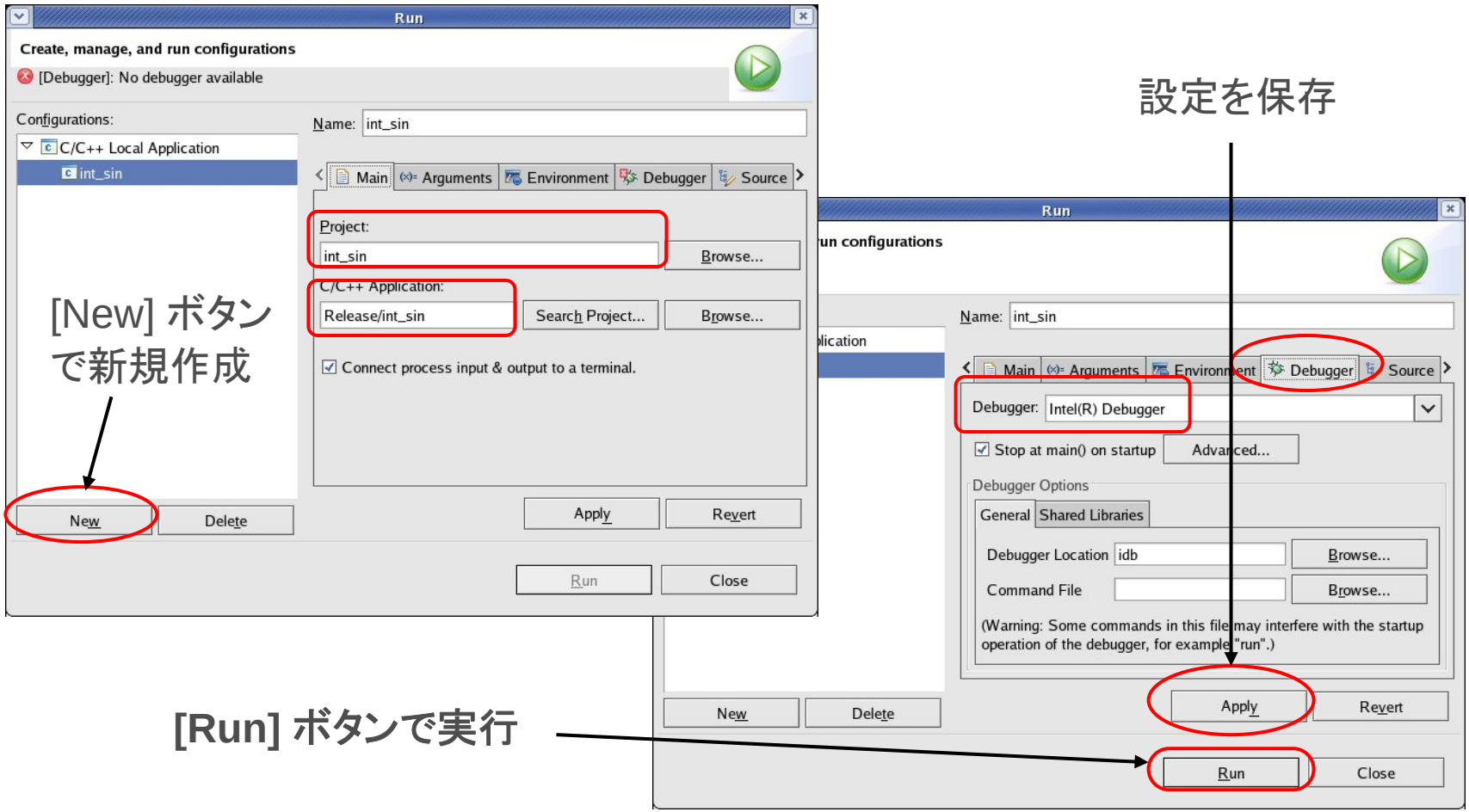
プロジェクトのビルド

- [Project] – [Properties] からビルド構成（Debug/Release）を指定し、[Project] – [Build Project] よりビルド開始



実行構成の作成および実行

- [Run] – [Run…] から実行構成を作成



本セミナーの内容

1. インテル® コンパイラーの特長

2. 開発環境

3. インストール手順

4. コンパイル

5. 最適化オプション

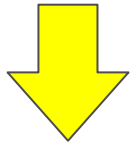
最適化オプション

インテル® コンパイラーには数多くの最適化オプションがあります。
以下に代表的な最適化オプションの一部をご紹介します。

オプション	説明
-O3	-O2 の最適化に加えて更に強力な最適化オプション
-x{W|N|B|P|T} -ax{W|N|B|P|T}	SIMD 命令による自動ベクトル化オプション x ー プロセッサー固有の最適化オプション ax ープロセッサー固有 ＋ 汎用 IA-32 の最適化オプション W:インテル® Pentium® 4 および互換プロセッサー N:インテル® Pentium® 4 SSE2 および互換プロセッサー B:インテル® Pentium® M および互換プロセッサー P:インテル® Pentium® 4 SSE3、インテル® Core™ Duo および互換プロセッサー T:インテル® Core™2 Duo、インテル® Xeon® 5100 シリーズおよび互換プロセッサー
-ip -ipo	関数のインライン展開などを行う、プロシージャー間の最適化オプション。 ip ー 単一ファイル内でのプロシージャー間の最適化 ipo ー 複数のファイルにわたるプロシージャー間の最適化
-fast	以下のオプションを 1 つにまとめた速度重視の最適化オプション -ipo -O3 -no-prec-div -static –xP
-parallel	マルチスレッド・コードを自動生成する並列化オプション

最適化ソリューション

インテル® コンパイラーを使用してもパフォーマンスが向上しない?



- その他いろいろなインテル® コンパイラー最適化オプションを試す。
  (**最適化ガイド**:http://jp.xlsoft.com/documents/intel/compiler/qr_guide_jp.pdf)
- コンパイラー・レポート・オプションを使用して最適化状況を確認する。
- インテル® VTune™ アナライザーを使用して、hotsopt を見極め効率的な最適化作業を行う。
- インテル® IPP、MKL などの最適化済みライブラリーを使用する。
- インテル® スレッディング・ツールで、マルチスレッド処理の動作を確認する。
- インテル® ソフトウェア開発製品セミナー(旧称: インテル® ソフトウェア・カレッジ)に参加して、インテル® プロセッサーのアーキテクチャーおよび最適化プログラミング知識を習得する。