



# **understand 用户使用指南**

V2.5  
2010.7



---

本文档提供的相关信息可能随时更改，恕不另行通知。Scientific Toolworks 不保证本文档不出现错误，也不为错误导致的后果承担责任。

权利声明：本文档的使用、复制或者正式出版必须遵循 DFAR 252.227-7013 (48 CFR) the Rights in Technical Data and Computer Software 章中(c)(1)(ii)节阐述的约束条款，承包商（制造商）为 Scientific Toolworks, Inc., 230 N 1680 E, Ste. OP1, St. George, UT 84790。

声明：对于获取到与本软件使用相关的租赁合约或者许可条款，公开使用、复制或者发布必须遵循 FAR 52.227-19 Commercial Computer Software-Restricted Rights 章中(c)(1)和 (2)节阐述条款。

产品编号：USTAND2.5-GEN-UG-522 (7/10)

Shiningstone provided

---

## 简介

本章介绍 Understand 2.5，本手册需要用户对自己项目所使用的编程语言有适度了解。

本章包含以下内容

什么是 Understand 2.5

许可核发

支持的语言

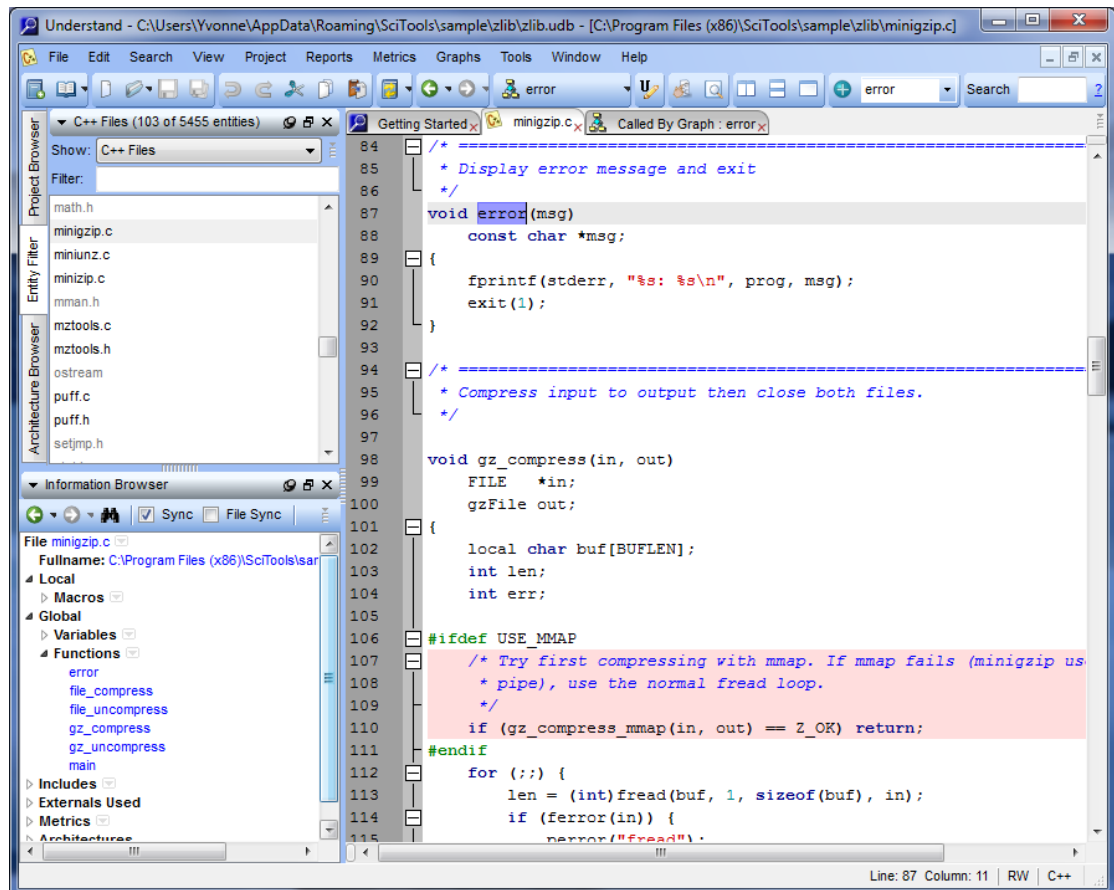
写给那些不愿意读手册的人

Shiningstone provided

## 什么是 Understand 2.5

Understand 2.5 是一款跨平台，支持多种语言，以软件维护为目标的交互式开发环境，主要有助于大型项目的新编和历史代码的理解和维护，支持的语言包括 Ada, C++, C#, FORTRAN, Java, JOVIAL, Delphi/Pascal, PL/M, VHDL 以及 Web 语言。

本软件通过详尽的交叉引用，语法着色“智能”编辑器和多种形式的逆向工程绘图提供代码导航功能。



Understand 2.5 为软件项目创建依赖关系和结构的仓库，通过这个仓库用户能够更快捷地了解代码。

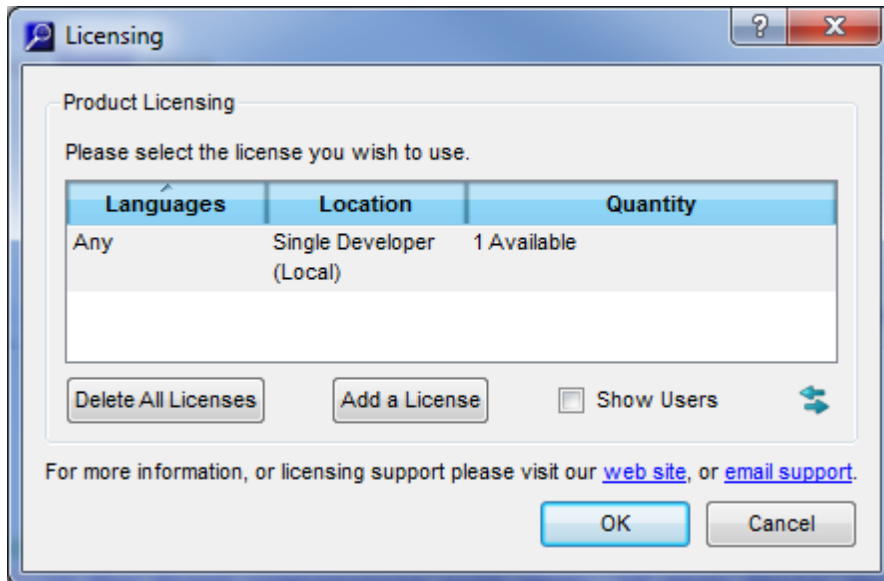
基于 Understand 2.5 的分析特性，用户能够更迅速的了解到以下问题：

- 这个实体是什么？
- 这个实体改变了什么？
- 这个实体引用了什么？
- 这个实体被谁依赖？
- 这个实体依赖谁？

基于 Understand 2.5 的结构特性，用户能够建立起代码单元的层次结构，对代码单元命名，灵活选择合适的方式进行操作，使得这个结构能够从更符合用户关注的角度对代码进行分析。

## 许可核发

Understan 2.5 的标题栏显示当前获得的许可类型。



如果用户拥有多个许可，在这里可以选择期望使用的许可类型。当获取到一个新的许可认证，点击 **Add a license**，然后选择评估版，个人版或者 license 服务器的名称。

如果用户使用的许可类型为非固定许可，可以通过点击 **Show Users** 查看当前激活的用户，旁边的双箭头符号可以刷新许可使用信息。

## 支持的语言

下面的列表简单介绍了 Understan 2.5 支持的语言以及编译器版本：

- Ada: Understan 2.5 支持 Ada83, Ada95 以及 Ada05。
- C/C++: Understan 2.5 支持 K&R 和 ANSI C 以及 C++语言的大部分特性，兼容大部分流行的 C 编译器，不过 C++模板特性 Understan 2.5 尚未支持。
- C#: Understan 2.5 支持 C#。
- FORTRAN: Understan 2.5 支持FORTRAN77, FORTRAN90和FORTRAN95的自由格式和固定格式，还支持Harris FORTRAN以及DEC FORTRAN。本软件关注对FORTRAN通用的编译器扩展，用户使用中发现的不支持的编译器扩展，可以通过support@scitools.com联系我们。
- Java: Understan 2.5 支持 JDK 1.3/1.4/5/6，不过 JDK 5 引入的 generics 暂不支持，包含 generics 的源代码能够被解析，但是相关信息会被忽略。
- JOVIAL: Understan 2.5 支持 JOVIAL73 和 JOVIAL3。
- Pascal: Understan 2.5 支持Borland's Delphi和Borland's Turbo Pascal的全部版本，同时支持ISO 7185: 1990 (也称为Unextended Pascal) ，并且可选支持Ingres内嵌的SQL语法。
- PL/M: Understan 2.5 支持 PL/M 80/86 标准版本。
- VHDL: Understan 2.5支持VHDL-87, VHDL-93和 VHDL-2001。
- Web: Understan 2.5支持HTML, PHP, CSS以及Javascript。

对于某种特殊语言的语法的支持信息，用户可以通过访问 Scientific Toolworks 的网站上的 build 日志

(<http://www.scitools.com/support/buildLogs.php>) 和论坛

---

(<http://www.scitools.com/support/forum/>) 进行查询。

## 写给那些不愿意读手册的人

我们可以想象，甚至支持用户跟 Scientific Toolworks 的大部分工程师一样，一款新的软件一到手就开始自己尝试使用，所以您可以把这份手册仅仅当成是一个安全网，或者只在必要的时候拿来查看一些并不那么明显的特性。然而，我们还是强烈建议在您将这份手册扔入粉碎机之前，浏览一下本章节，以获取一些能够有效使用 Understand 2.5 的小方法。

下面是一些不使用本手册也能解决 Understand 2.5 使用过程中的问题的方法：

- 使用 **Getting Started** 视图中的链接（菜单栏上 **Help->Getting Started**）。
- 选择菜单栏上 **Help->Help Content**。
- 使用 **Help->Example Projects** 对样例代码进行演练。
- 选择 **Help > Frequently Asked Questions** 查看本公司网站上的 FAQ
- 选择 **Help > View SciTools Blog** 查看本公司 blog，点击 **Refresh** 按钮可以显示最近更新的主题。

对于更高级的用户，可以从以下地方获取进一步的信息：

- 选择 **Help > About Understand** 检查当前使用的版本信息。
- 通过 <http://www.scitools.com/support/buildLogs.php> 检索 build 日志，该网页上的“Sign up to receive via Email”，可以让您以邮件方式获取到最新的 build 信息。
- 通过 <http://www.scitools.com/support/forum/> 访问本公司论坛进行提问。
- 选择 **Help > Key Bindings** 获取 keystroke 方面的帮助。
- 查看 <http://www.scitools.com/documents/metrics.php> 获取 metrics 方面的细节信息。
- 通过 **Help > Perl API Documentation** 获取脚本方面的帮助。

---

## 部件和术语

本章介绍 Understand 2.5 基本窗口, 通过阅读本章内容, 用户能够更快更容易地使用本软件。  
本章包含以下内容:

Understand 2.5 窗口介绍

Understand 2.5 术语

启动 Understand 2.5

无处不在的右键菜单

快速查找代码

信息浏览器

源代码编辑器

结构浏览器

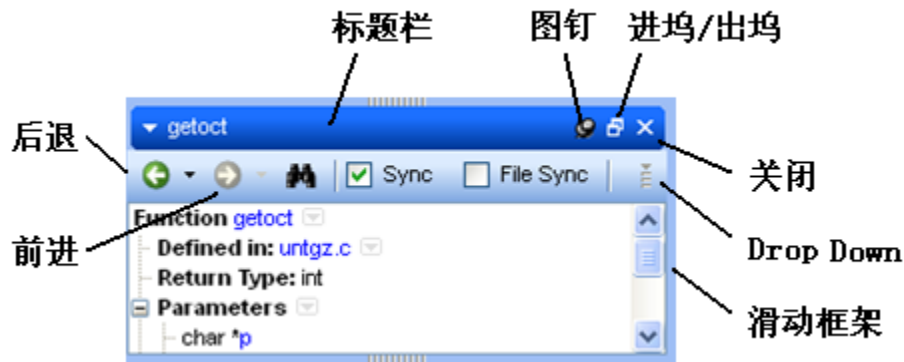
视图

ASCII/HTML 格式报告


报告定制相关的 Perl/C API


## Understand 2.5 窗口介绍

Understand 2.5 拥有一个主窗口和许多可以在应用窗口中打开的小窗口，这些小窗口可以按照用户习惯放置。



标题栏：点击标题栏可以将小窗口拖动到主窗口范围内的任意位置，当小窗口位于主窗口边缘，会出现船坞区域的提示，小窗口放在这里，将会自动固定到船坞区域。

图钉和抽屉：点击图标  使得船坞区域的小窗口成为主窗口的一个标签页，这个标签页像抽屉一样活动——将鼠标移到标签页的标题，抽屉自动弹出，类似的，将鼠标移出抽屉范围，抽屉会自动关闭。

抽屉左上角位置的  可以使抽屉固定打开，打开的抽屉如上图所示。

进坞/出坞：点击  更改窗口的船坞状态。

关闭：点击  关闭窗口。

Drop-Down：点击  展示关系区域的右键菜单，

Right-clicking an item within an area usually displays a right-click menu specific to that item.

滑动框架：滑动框架可以随意调节相邻窗口的相对大小。

前进/后退：标题栏在不同类型区域有不同的图标，在信息浏览类的窗口，可以通过前进/后退可以查看曾经浏览过的内容，对于其他类型的窗口，展示的图标种类有可能不同。

## Understand 2.5 术语

了解本章介绍的 Understand 2.5 约定的一些术语，能够更充分的理解本手册的内容以及更有效的与技术支持团队进行邮件或者电话沟通。

层级：层级表示代码单元（或者实体）组成的层次结构，可以由用户手动创建，也可由本软件自动生成。一个层级可以不完整（例如一个层级的扁平化扩展有可能不会关联数据库中的所有代码实体），也可能不唯一（扁平化扩展的层级可能不会处理其预设属性）。

数据库：代码经分析后产生的中间结果，以及工程设置保存在数据库，其缺省扩展名为“.udb”。

实体：Understand 2.5 描述的“实体”表示任何包含信息的事物，具体来说，代码中声明或者使用的 **标识**、包含工程的文件、子程序、变量、源文件都可以被称为实体。

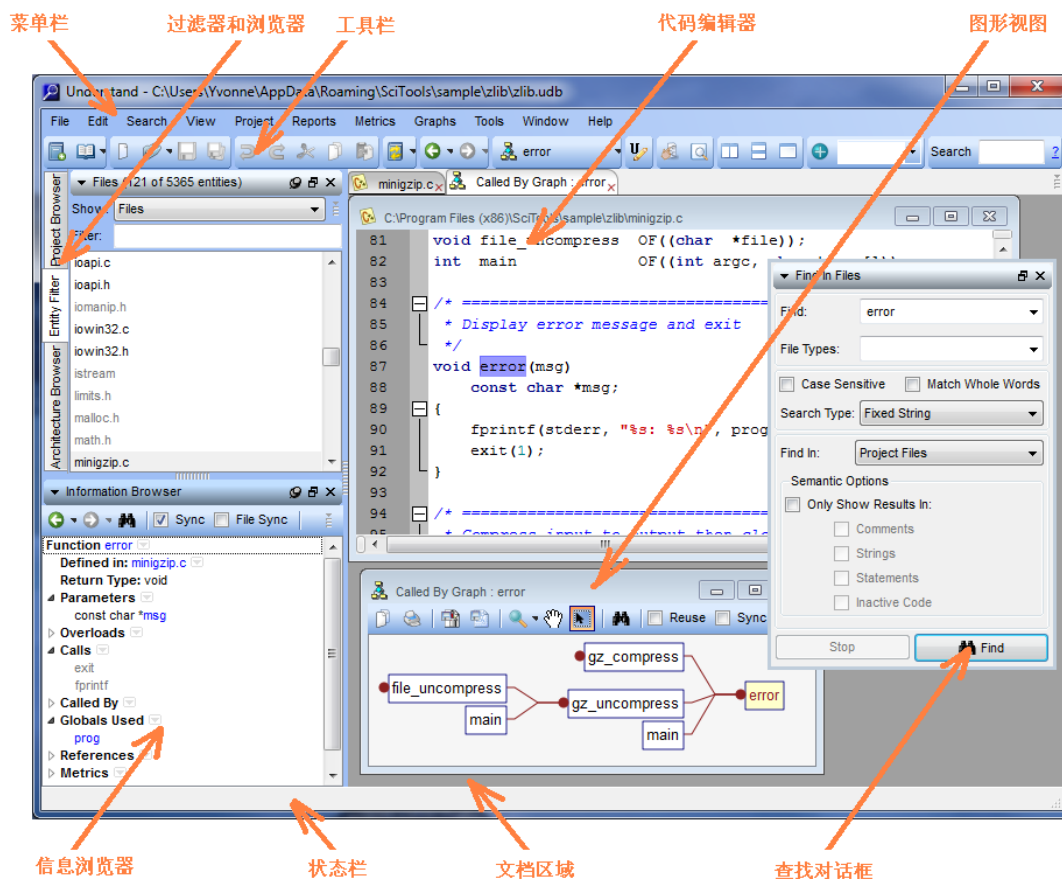


工程：表示源代码的集合以及相关的配置和参数，工程文件包含源文件清单和工程设置。

关联：相互作用的实体之间的关系，关联的名称来源于编程语言的语法和语义，例如过程式实体具有“调用”和“被调用”的关联对象。

脚本：通常指 perl 脚本，脚本可以通过 Understand 2.5 的图形用户界面或者外部的脚本命令执行。Understand Perl API 提供了快捷的访问 Understand 数据库所有信息的接口。

部件：下面的图形展示了一些 Understand 2.5 图形用户界面中常用的部件。



## 启动 Understand 2.5

在 Windows 系统安装完本软件，启动本软件的命令自动在 **Start** 菜单的 **SciTools** 目录下添加。

启动 Understand 2.5 后，窗口内可以看到 **Getting Started** 标签页，点击 **New Project...** 创建一个新的工程，更详细的信息在[创建一个新工程](#)中介绍。

最近使用过的工程在 **Getting Started** 标签页中显示，点击可以打开，如果需要打开未列出的工程，可以点击 **Open Project...** 查找。

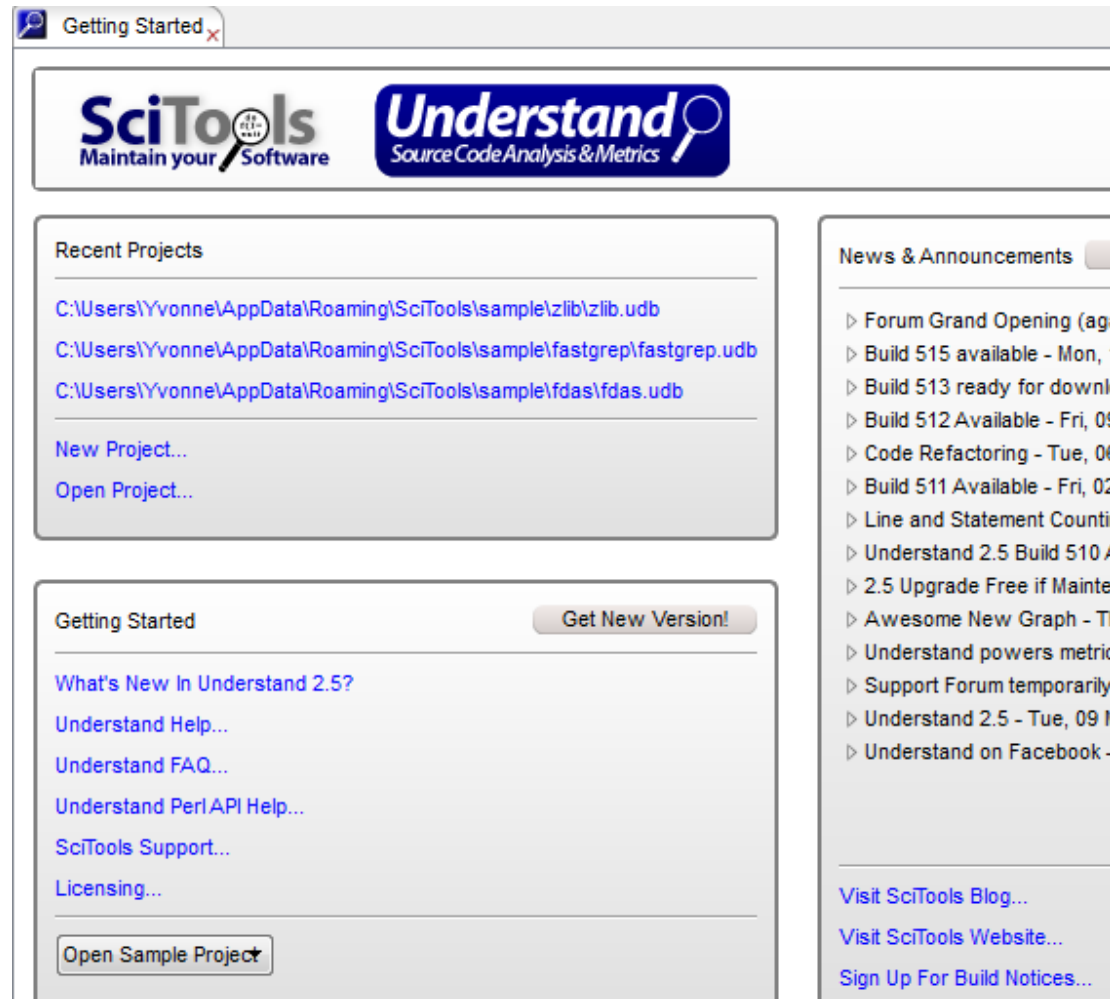
通过 **File > Open > Project** 和 **File > Recent Projects** 也可以打开一个工程。

对于刚开始接触的用户，可以使用 **Getting Started** 内的链接进行学习。点击 **Open Sample Project**，选择使用与当前项目使用的语言一致的示例工程进行学习可以加速对本软件的了解。

对于熟练使用本软件的用户，使用 **News&Announcements** 内的链接可以获取本软件的一些最新信息。

菜单栏的 **Help>Getting Started** 可以将关闭的 **Getting Started** 标签页重新打开，取消 **Show**

On Startup 勾选可以避免该标签页在每次启动本软件后自动打开。



当前工程相关的工作结束，可以打开另一个工程或者使用 **File > Close <project\_name>.udb** 关闭该工程。如果某些文件发生了改变，可以选择保存这些变动或者针对每个文件执行放弃保存的操作。

菜单栏上的 **Help > Check for Updates** 可以帮助用户确认当前使用的版本是否是最新版本（如果存在更新的版本，Getting Start 标签页会显示一个 **Get New Version** 的按钮）。

## 启动 Understand 2.5 的其他途径

通过命令行启动 Understand 2.5 的方法可以参考第 13 章 命令行操作。

Understand 2.5 为多用户提供了个性化的初始文件，能够保存每位用户的惯用选项，初始文件的查找顺序设定为：

- 1%APPDATA% (local settings and applications)
- 2%WINDIR% (typically c:\Windows)
- 3%STI\_INIDIR%

## 无处不在的右键菜单

Understand 2.5 的右键菜单为用户提供非常广泛的快捷功能，通过右键菜单可以即时获

取相关信息，通过弹出菜单进行一些快捷操作。

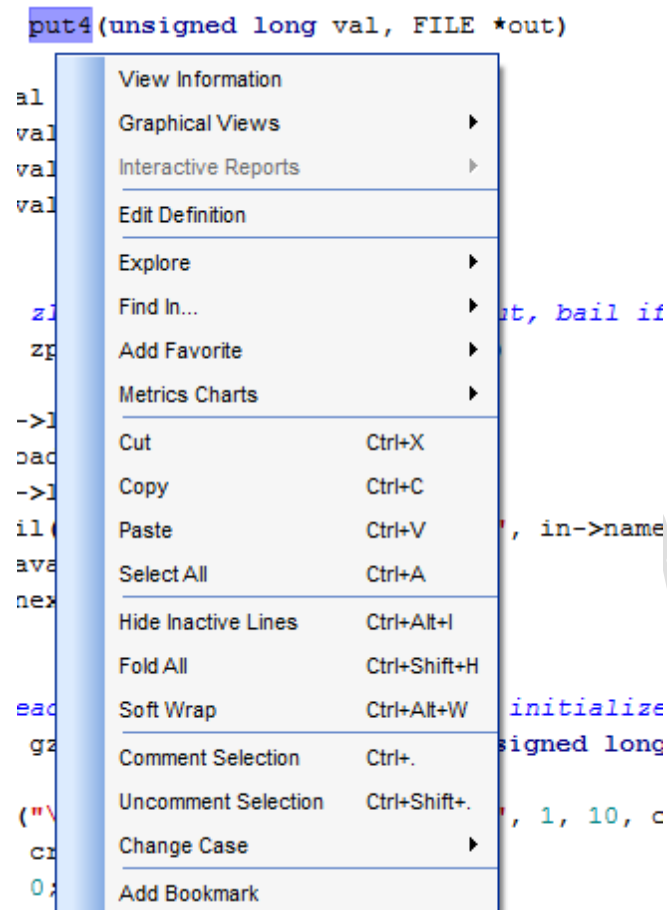
小技巧：

保持 **ctrl** 键按住不放同时单击鼠标右键可以创建新的窗口。

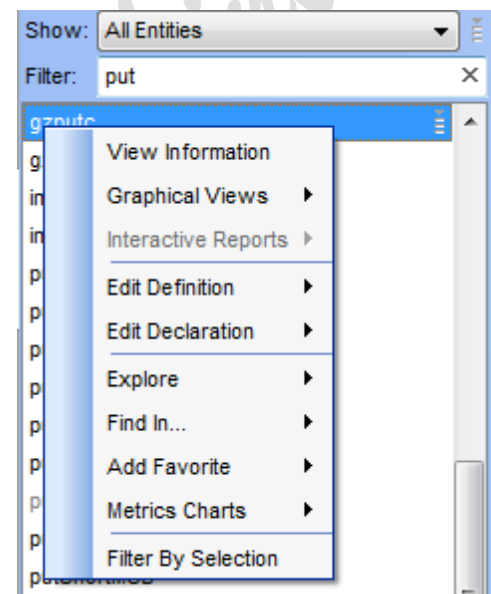
任何实体上都可以尝试单击鼠标右键获取进一步的相关信息。

示例：

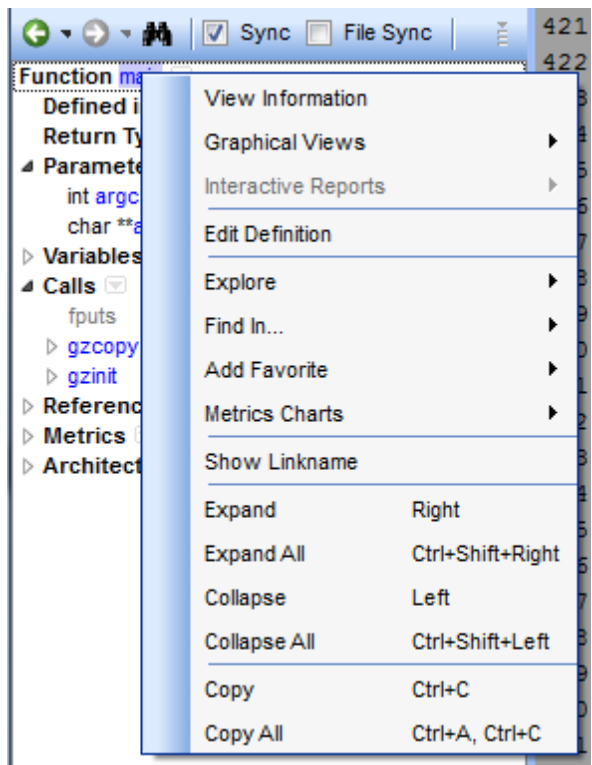
代码编辑器内的实体的右键菜单



过滤窗口内的实体的右键菜单



信息浏览器内实体的右键菜单



## 快速查找代码

Understand 2.5 为代码快速定位提供了多种途径，相关窗口包括过滤窗口，实体定位器和查找对话框。

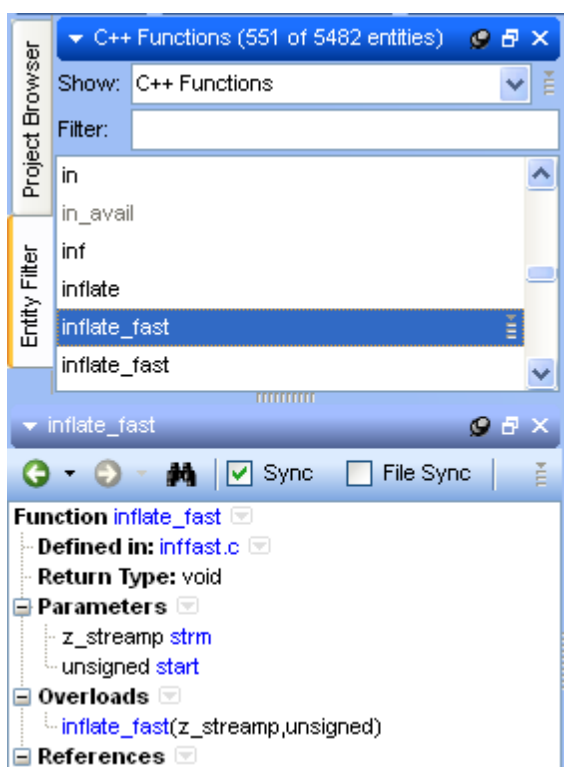
## 实体过滤器

Understand 2.5 提供的过滤窗口将数据库中的内容按照类别分为文件，类，函数，对象，类型，宏，子程序，包，模块，块，方法，接口，SQL 表等视图，通过这些视图可以实现快速查找。视图分类依赖于对 Understand 2.5 的语言设置。

激活过滤窗口，输入一个字符可以快速显示出当前视图以该字符开头的实体。

默认情况下，信息浏览器会显示被选中实体的所有相关信息，妥善使用这些信息是有效使用 Understand 2.5 快捷查找功能的关键。

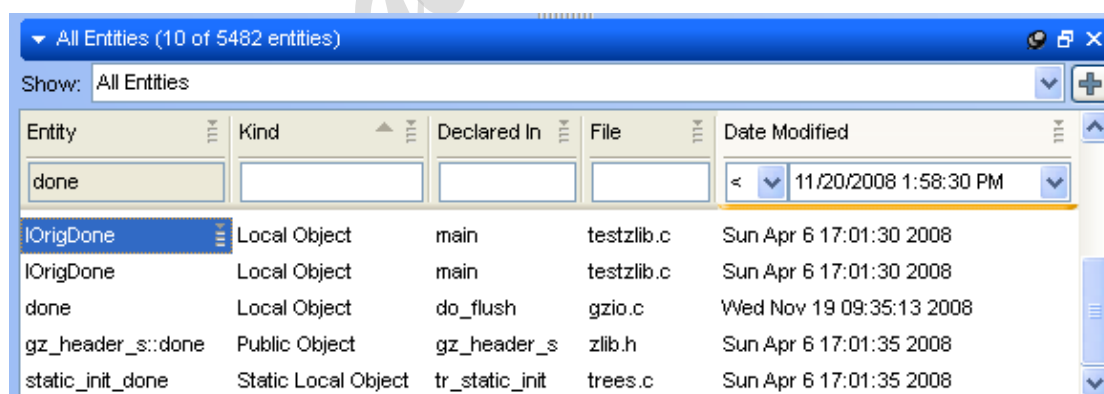
实体过滤器和信息浏览器的详细信息，可以参见实体过滤器和信息浏览器。



## 实体定位器

过滤器提供了查找当前工程声明和使用的实体的快速查找，但是，有一些实体，如局部参数，变量和未识别变量（代码中使用但是未声明的变量）不在查找列表范围内。使用实体定位器，可以查找工程的整个数据库。

使用 **View > Entity Locator** 打开实体定位器。



默认情况下，这个窗口显示当前工程的所有实体，使用每列上方的输入域，可以通过特殊字符甚至是正则表达式对实体进行查找。

实体定位器的详细介绍，参见实体定位器。

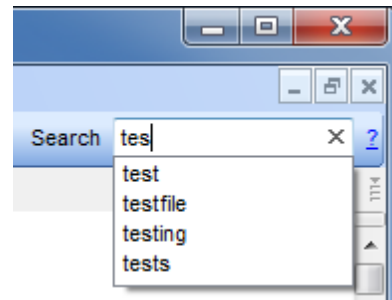
对于其他窗口，右键菜单也能够被激活。

选择多个行和列，可以拷贝其内容到剪切板，粘贴时，内容以tab进行分隔。

## 快速搜索

快速搜索实现了在成千上万行代码的工程中的完整搜索，输入内容被立即匹配而显示出来。

快速搜索提供了多种功能强大的选项，详细介绍在快速搜索。



## 文件内查找

Understand 2.5提供了与UNIX提供的grep命令类似的操作，在多个文件中查找一个指定的字符串。通过Search菜单或右键菜单中的选项，均可激活文件内查找对话框。

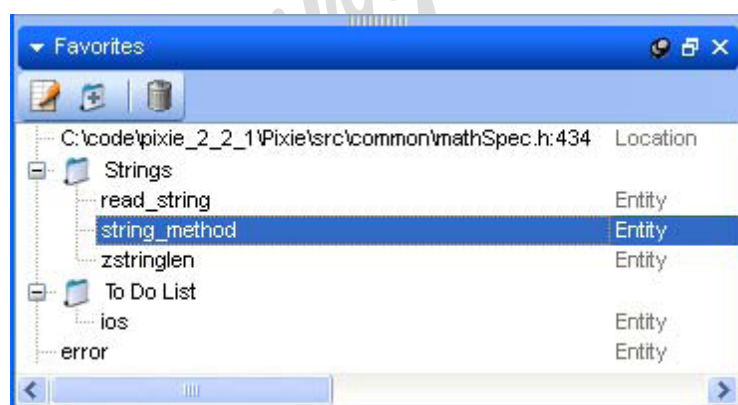
点击Search图标，查找结果窗口显示所有与指定字符串或者正则表达式匹配的出现点，双击其结果显示相关代码窗口。

Find Options可以选择大小写匹配和模式匹配。

文件内查找在文件内查找详细介绍。

## 常用内容

经常使用到的实体和代码可以保存到常用内容列表。右键菜单中 **Add Favorite** 将一个选中内容添加到列表，**View>Favorite** 打开常用内容列表窗口，双击选择项即可跳转到对应位置。



常用内容，参见常用内容。

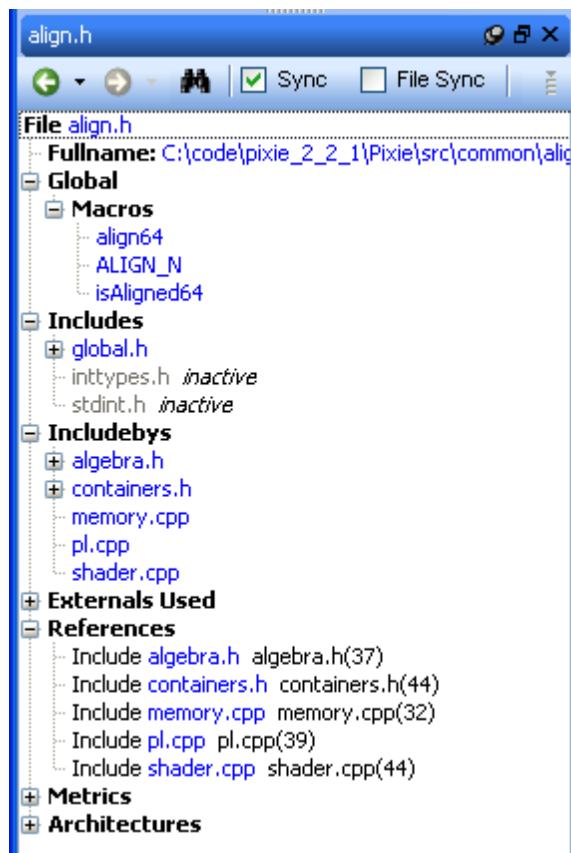
## 信息浏览器

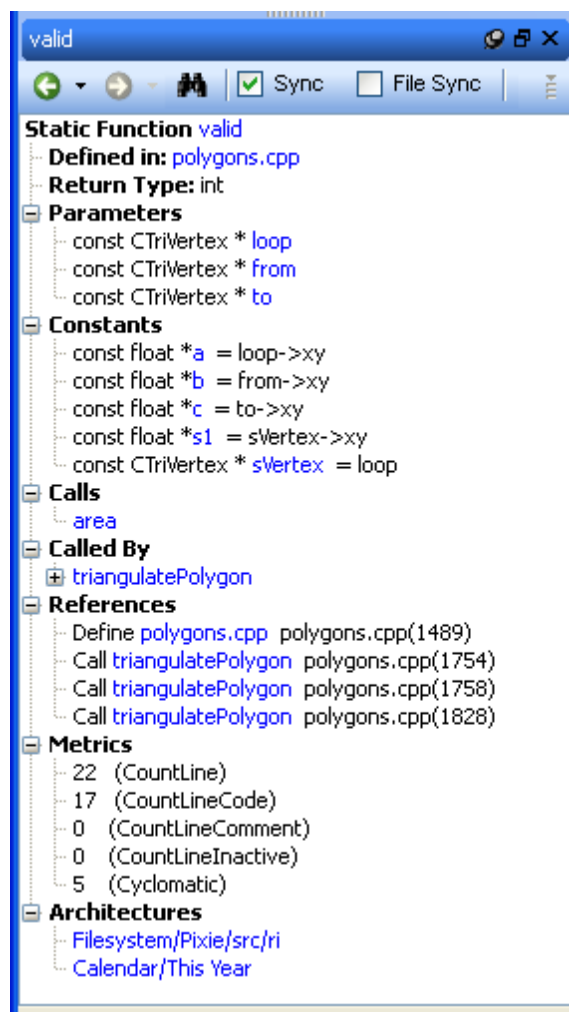
信息浏览器展示了Understand 2.5包含的代码的全部信息，所有类型的实体都可以用信

息浏览器进行信息查看。

不同类型的实体，包括源文件，类，成员，函数，类型，方法，包，接口等，在信息浏览器中展示的内容会有所不同。具有层级特性的信息（如调用关系）可以按照层级展开。

下面分别展示了文件和C函数的信息浏览器视图。



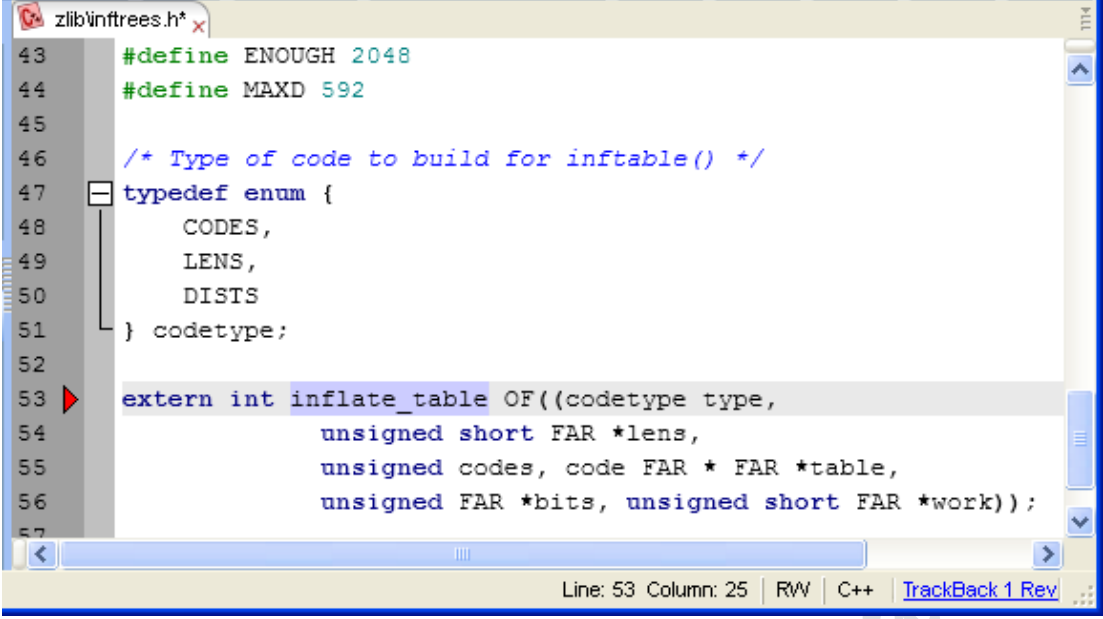


## 代码编辑器

Understand 2.5提供的代码编辑器不仅提供了基本的文本编辑功能，而且支持着色，提供被编辑代码的信息。

几乎任何地方进行双击都可以访问代码，使用工具栏上的Prev和Next图标可以在浏览历史的视图进行切换。





```
43 #define ENOUGH 2048
44 #define MAXD 592
45
46 /* Type of code to build for inflate() */
47 typedef enum {
48     CODES,
49     LENS,
50     DISTS
51 } codetype;
52
53 extern int inflate_table OF((codetype type,
54     unsigned short FAR *lens,
55     unsigned codes, code FAR * FAR *table,
56     unsigned FAR *bits, unsigned short FAR *work));
```

Line: 53 Column: 25 | RVW | C++ | [TrackBack 1 Rev](#)

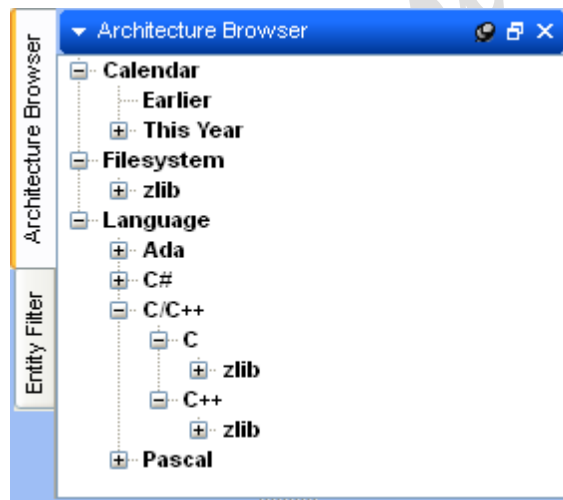
在编辑器中也能够使用右键菜单，任何地方使用右键菜单都可能获取到相关的进一步信息。

代码编辑器详见代码编辑器。

## 层级浏览器

层级浏览器提供层级管理功能，显示了所有在数据库中定义的层级结构的列表，为用户 提供操纵这些层级结构的方法。

下面的窗口展示了Understand 2.5提供的自动构建的层级信息：日程记录，目录结构，语言信息。这些分类进行了一些扩展，以便突出展示示例应用的顶层节点。



用户可以直接使用Understand 2.5提供的默认层级，创建自己的层级，导入和导出层级信息（保存为XML文件），针对层级的不同层次生成图和度量以及通过过滤整合层级。

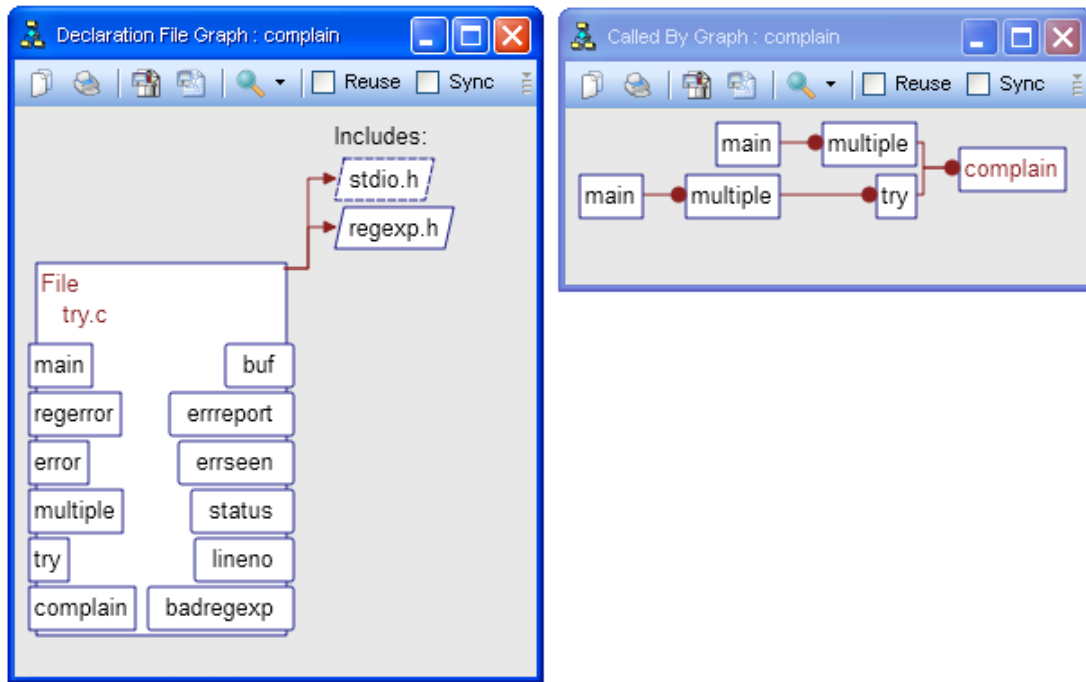
层级浏览器，参见层级浏览器。

## 图形视图

Understand 2.5 分析软件代码，创建一个包含所有实体以及实体之间相互关系的数据库，为这个数据库提供图形视图的浏览方式，这些视图分为三种类型：

层级视图展示实体之间的关系。每个视图基于实体的依赖关系（如“调用”）从指定实体开始依次显示之后的后续实体

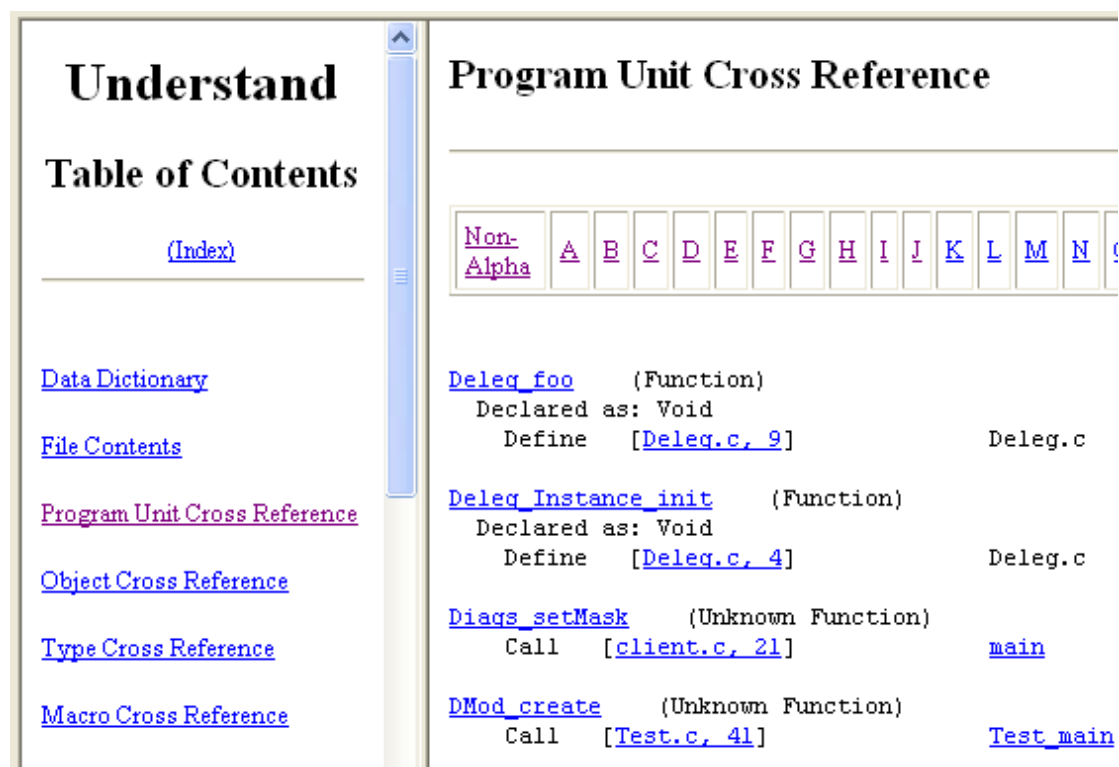
结构视图展示软件结构中所有实体（如包，函数，过程或者任务）的结构信息



图形视图的详细信息，参见使用图形视图。

## ASCII/HTML 格式报告

Understand 2.5 的视图展示了单个实体的相关信息，ASCII/HTML 格式报告则将所有实体的相关信息汇聚在一起。



ASCII/HTML 报告还能够显示一些不支持交互操作的信息，如工程度量和质量报告，这些信息非常适合用 web 浏览器来显示。

ASCII/HTML 报告详细信息，参见使用报告。

#### 报告定制相关的 Perl/C API

Understand 2.5 的数据能够直接被外部编写的程序和脚本访问，为此 Understand 提供了 C API（C/C++ 或者能够使用 C 库的其他语言）和 Perl 接口。

通过 API，本软件开放了 GUI 和报告生成器的访问权限。

本手册不包含这些 API 的介绍，不过可以使用 **Help>PERL API Documentation** 获取进一步信息。

通过 **Reports > Project Interactive Reports** 和 **Graphs > Project Graphs** 可以显示一系列用户使用 Perl API 创建的插件。联系 support@scitools.com，获取创建插件的方法，SciTools 论坛 <http://scitools.com/support/forum> 和 SciTools 博客 <http://scitools.com/blog> 也包含了关于插件的信息。

---

## 工程设置

本章介绍创建一个新工程用于分析用户关心代码。

本章包含以下内容：

关于Understand 2.5工程

创建新工程

工程配置对话框

语言类别配置

文件类别配置

文件类型

文件选项

计划活动

度量

报告

Visual studio

Ada选项

C++选项

C#选项

FORTRAN选项

Java选项

JOVIAL选项

Pascal选项

PL/M选项

VHDL选项

设置惯用选项

代码分析

Understand 1.4工程转换

---

## 关于 Understand 2.5 工程

Understand 2.5与编译器类似，不过它生成的是代码相关信息，而不是可执行代码。

为了能够分析代码，Understand需要获取比编译器需要的更多信息，例如：

- 需要分析的源代码
- 源代码类型
- 标准库路径和头文件路径
- 为代码提供类的Java .jar文件所在位置
- 预处理需要的编译器或者环境特定的宏定义
- 应用相关的宏定义
- 执行参数（如整数精度）和列截断（column truncation）
- 名字空间

对于亲手开发出来或者已经维护过一段时间的项目，这些信息并不难提供。但是，对于新接手的代码，就可能不得不查阅大量的工程构造文件（如makefile）以获取这些信息来让Understand更精确地解析代码。

Understand 2.5提供了GUI来让客户能够更容易的构建和解析代码。

### Understand 2.5 工程数据库

Understand 2.5的工程数据库使用自定义的二进制格式进行存储，这种存储方式使用了网络/对象格式，优化了Understand 2.5的信息存储。

Understand 2.5的数据库文件使用扩展名.udb。

工程文件支持多进程读取，但是尚不支持多进程写入。

偶然情况下，Understand 2.5引入的新特性需要数据库格式的变更，这些变更在变更日志中发布。当Understand发生类似变更的升级，既有工程在打开的时候能够自动重新解析以适应这种变更。

### 创建新工程

为了分析代码，必须先创建一个工程，指定需要解析的源文件。Understand 2.5分析代码，创建可供浏览的数据库，这个数据库还能够通过GUI或者命令行工具进行增量更新。

本节介绍如何创建一个新的工程，新的工程被存储到以udb为扩展名的工程数据库。

创建工程包括以下步骤：

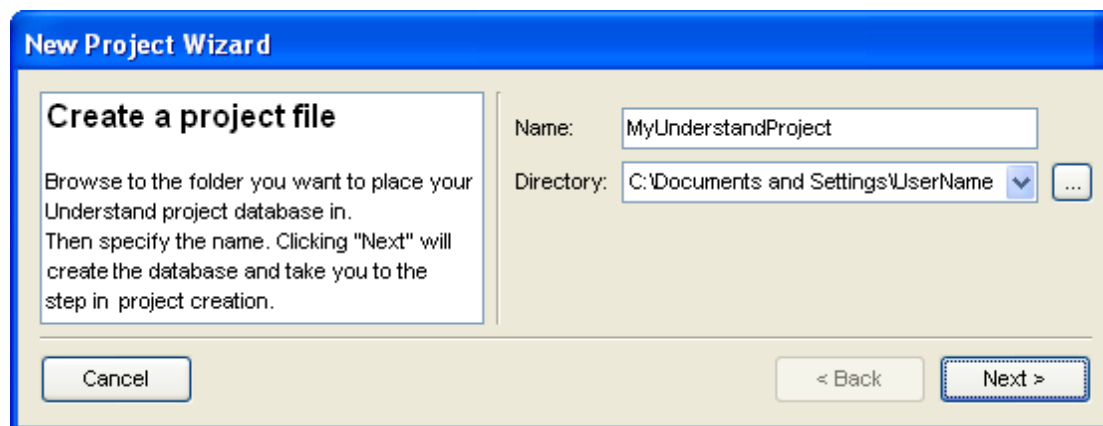
- 1 在Getting Started标签页点击**New Project**链接，或者在菜单栏选择**File>New>Project**。
  - 默认情况下，这个操作打开一个新工程引导程序（详见）
  - 如果选择了禁用引导程序，这个操作将会打开一个“Create new project as...”对话框，确定期望创建工程数据库的路径，在File name输入域输入工程名称（Understand自动添加.udb扩展名），点击Save，随即弹出Understand工程配置对话框（详见）。

### 新工程引导程序

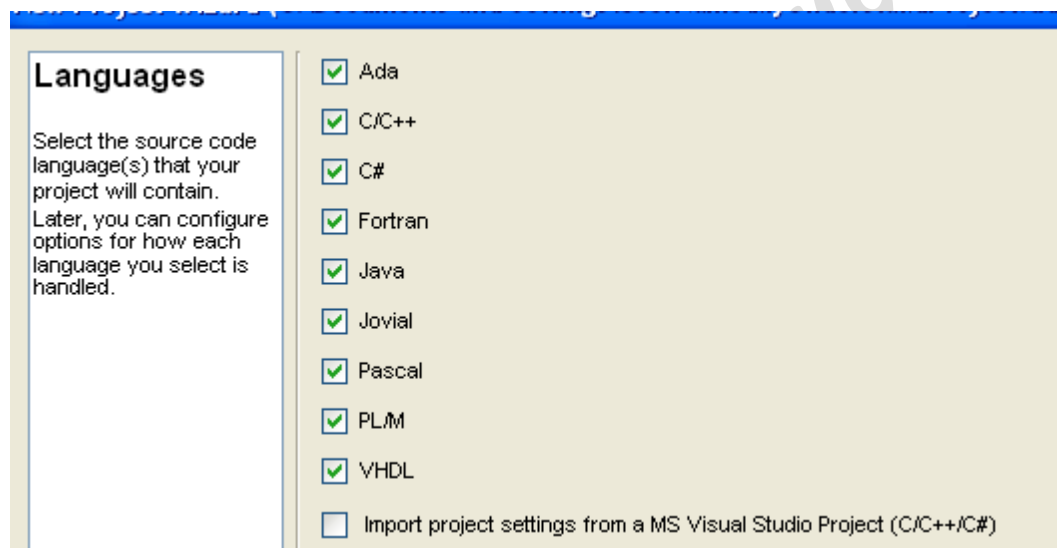
在没有禁用新工程引导程序的情况下，这个引导程序负责帮助用户创建一个新的工程。在

Getting Started标签页点击**New Project**链接，或者在菜单栏选择**File>New>Project**激活该引导程序。

1 在引导程序Create a project file页面，输入工程名称，指定工程文件保存的路径。推荐将工程文件保存到源代码所在目录的顶级目录。如果指定路径不存在，会弹出提示窗口，向用户询问是否需要创建该目录。



2 点击Next，跳转到Language页面。



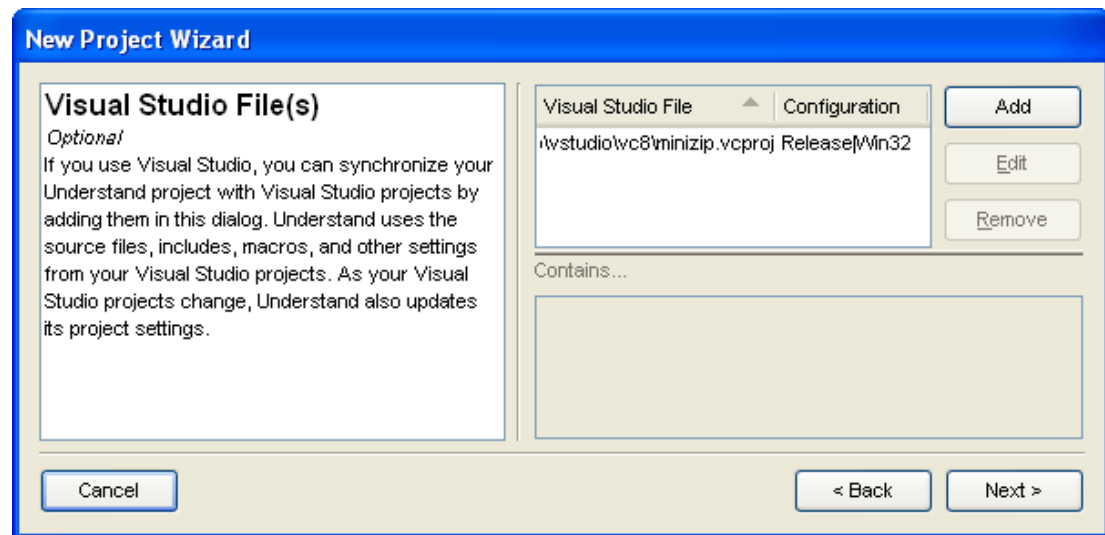
3 通过页面上的多选框为工程选择当前使用语言（详见语言类别配置），对于Microsoft Visual C创建的C/C++/C#工程，可以选择最下面的多选框直接导入工程设置。然后继续点击Next。

4 选择从Visual工程导入，可以看到Visual Studio File(s)页面，否则可以直接跳转到下一步。

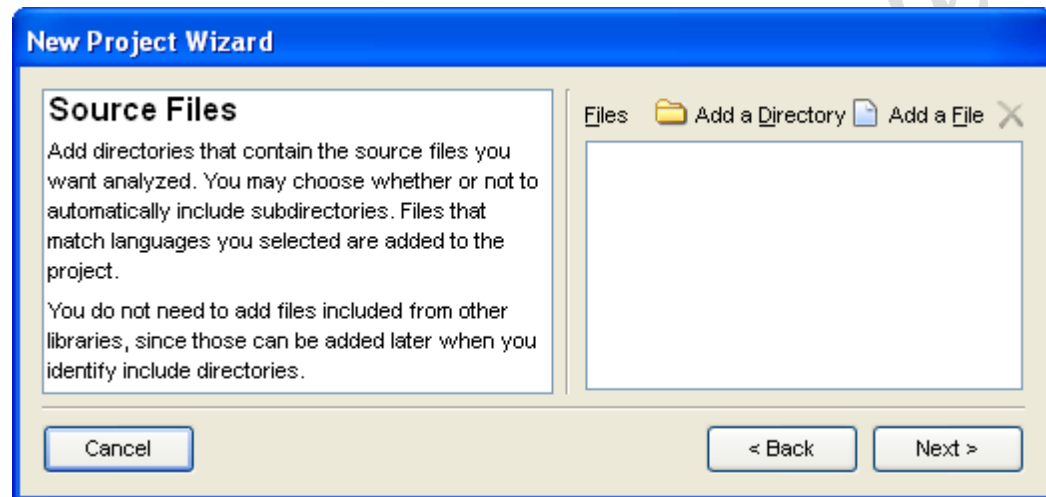
点击Add，弹出Add a new Visual Studio file对话框，可以将Understand工程与Visual Studio工程进行同步。在弹出对话框中点击...，确定Visual Studio工程文件，选择Understand分析代码需要遵循的工程设置，然后点击OK（Visual Studio详见）。

Understand支持增加多个Visual Studio工程，也可以通过Edit按钮对工程设置进行修改。

点击Next。



5 在Source Files页面，通过Add a Directory和Add a File向工程添加源文件。

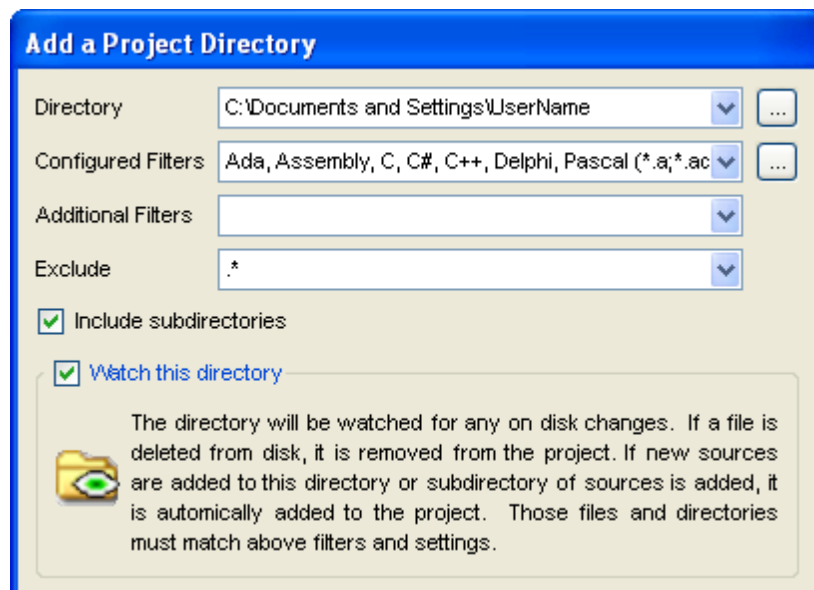


增加一个文件，只需找到这个文件然后添加。

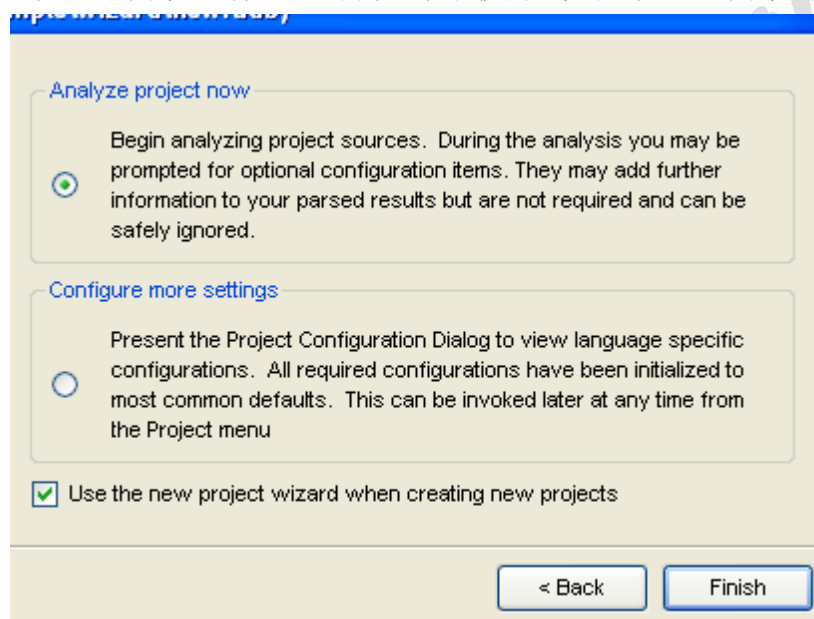
增加一个目录时，通过Directory定位目录，通过Configured Fileters修改需要添加的源文件类型，通过Additional Filters设置额外的过滤条件，通过Exclude直接排除不希望添加的文件类型，通过Include Subdirectories复选框选择是否需要递归添加，还可以通过Watch this directory选择是否对当前目录进行监视（详见）。

如果之前选择的是导入Visual Studio工程，新工程引导程序会自动添加相关文件，并在Source Files页面显示。

点击“X”按钮可以将选中的文件或者目录（包括其子目录）从工程中删除。



6 选择Analyze project now或者Configure more settings。选择Configure more settings，将会弹出工程配置对话框（详见）。这两个选项不影响后来的工程配置对话框的打开。



## 工程配置对话框

创建新工程或者选择 Project>Configure project 都可以打开工程配置对话框。

对话框左侧将代码分析时用到的一些工程设置进行分类列出，其中包括：

语言：指定使用语言（详见）

文件：指定待分析文件所在位置（详见）

文件类型：设置如何处理文件类型以及使用的文件扩展名（详见）

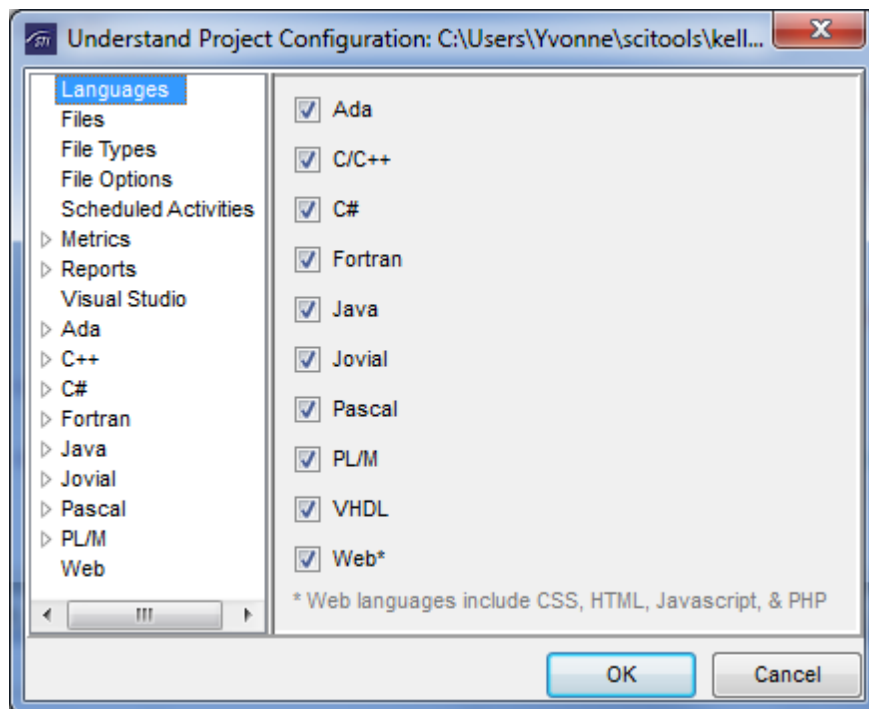
文件选项：设置文件编码和编辑风格（详见）

计划活动：设置定期执行的操作（详见）

度量：设置关心的度量指标（详见）

报告：设置关心的报告内容（详见）



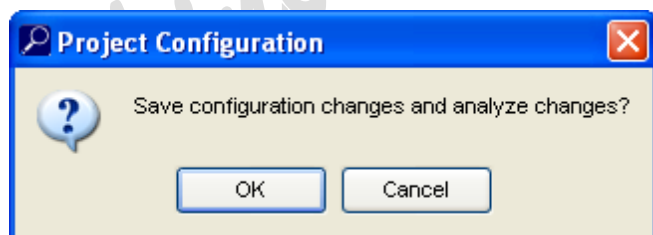


Visual Studio: 指定 Understand 工程同步的 Visual Studio 工程

语言特定选项: 为语言类别配置中指定的语言类型进行特性设置, 详见

- Ada选项
- C++选项
- C#选项
- FORTRAN选项
- Java选项
- JOVIAL选项
- Pascal选项
- PL/M选项
- Web选项

设置完工程配置后, 点击 OK 进行保存。对工程配置文件进行的任何修改, 包括创建新工程, 都会弹出告警对话框, 提示文件发生修改。



点击 OK, Understand 2.5 按照新的配置重新分析代码。

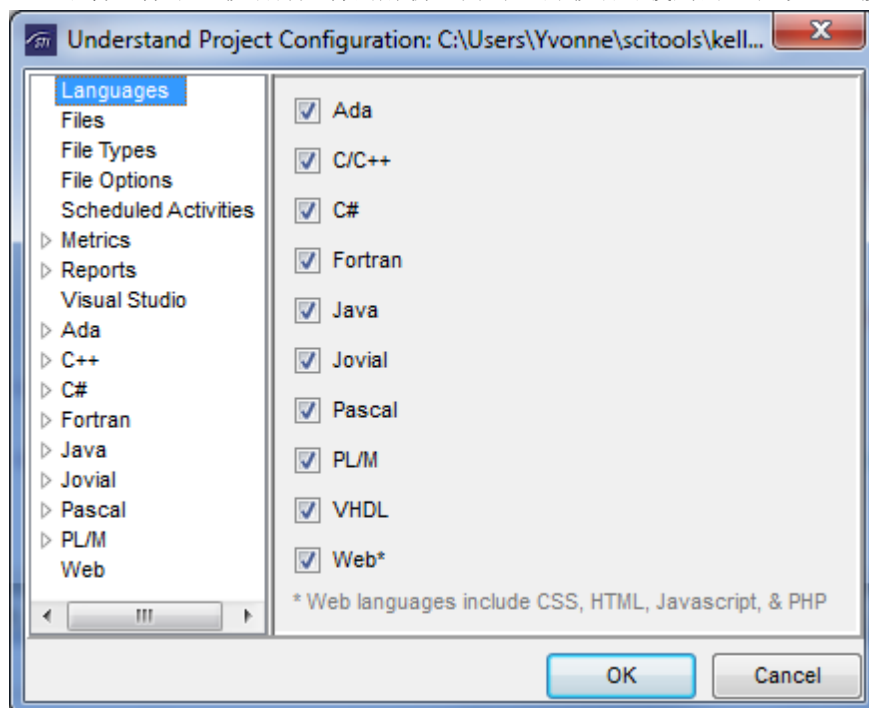
如果放弃修改, 点击Cancel, 然后在弹出的询问是否确定放弃修改的对话框点击Yes即可。

## 语言类别配置

工程配置对话框的语言类别配置中, 可以指定当前工程使用的语言种类, 语言种类不限一种。

每选中一种语言, 工程配置对话框左侧列表会增加对应的一个语言分类。

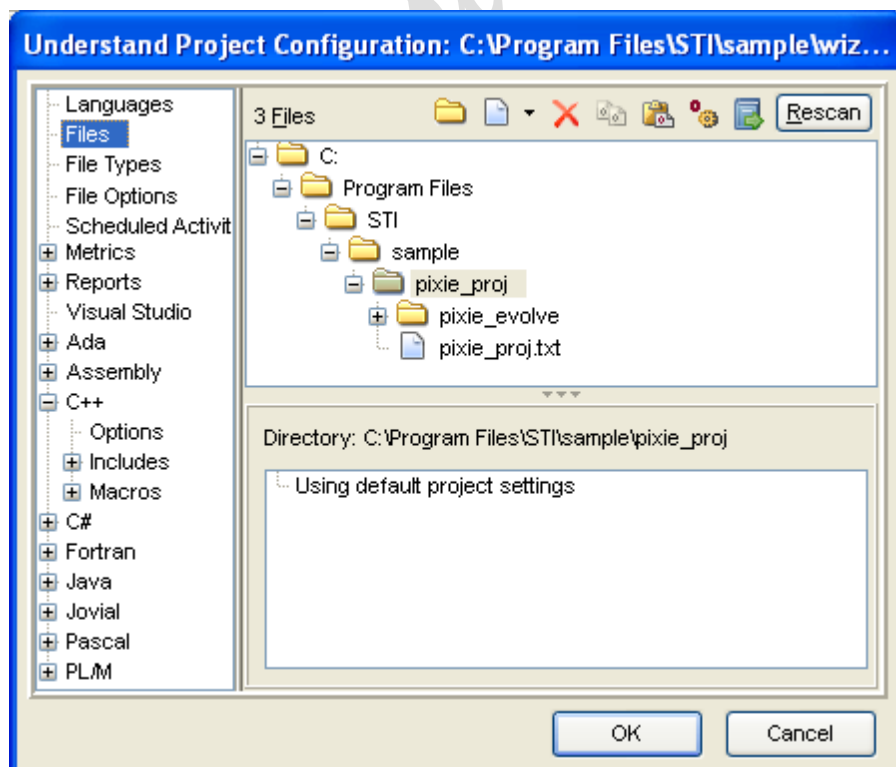
语言选择不止影响源文件的解析，同时也会影响可使用的过滤设置，度量和报告。



如果选择了多种语言，不同语言之间的相互依赖也能够被 Understand 识别，例如，使用 C 语言编写的代码调用 Java 的方法，这种依赖关系也能够体现在 Understand 的数据库中。

## 文件类别配置

通过工程配置对话框的文件类别配置项添加源代码目录和文件，删除指定文件，以及针对指定目录或者文件对语言相关选项进行配置。











除此之外，通过将Understand工程与MS Visual Studio工程文件绑定也能够添加待分析文件（详见Visual Studio）。

在对话框上方，一个可以展开的树形结构显示了所有已添加目录和文件，此外文件个数也同在对话框上方显示。


下方显示选中目录或者文件对一些工程配置相关选项的重新设置。

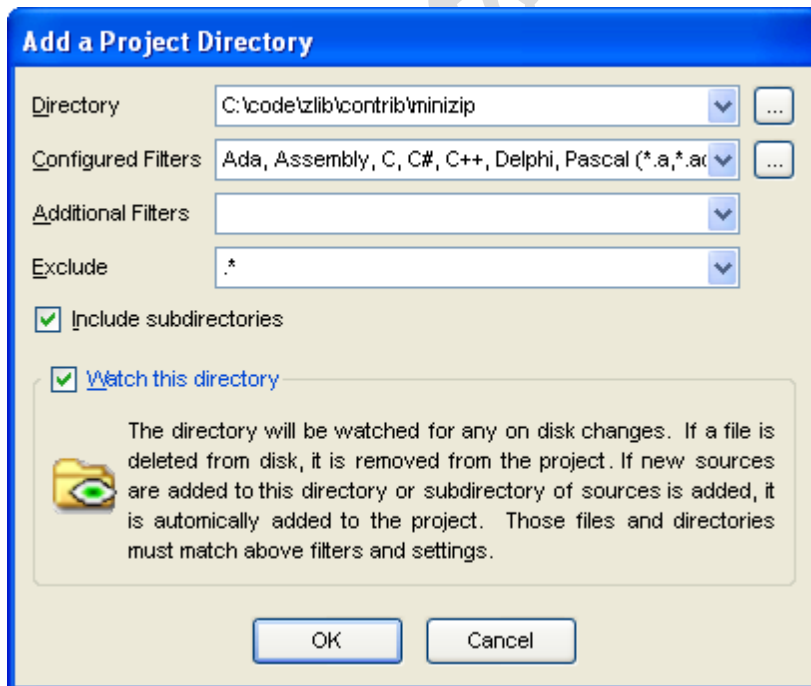
对话框上方图标功能如下所示：

-  打开添加目录对话框
-  打开添加文件对话框
-  选择添加文件或者导入文件列表
-  从工程中删除选中目录或文件
-  复制选中目录或文件的配置
-  粘贴配置
-  配置选中目录或文件
-  设置目录的可移植选项

所有修改在点击OK后才会保存，如果添加的文件没有在树形结构显示，可以点击Rescan进行刷新。

## 添加目录

需要将某个目录加入工程，点击，打开目录添加对话框：



1 在Directory输入域输入目录全路径，或者可以点击...按钮打开文件浏览器选定期望的目录，然后点击OK。

2 在Configured Fileter域，点击...可以对其中的列表项进行添加和删除。在文件类型配置对



对话框选择过滤器，选择当前工程涉及的语言类型。请注意有一些Understand支持的语言没有在语言类别配置中显示出来，其中包括JavaScript，MSDos批处理语言，Perl，Tcl，Text和XML。

如果当前目录包含了扩展名没有列出的源文件，点击Configure，后续操作详见文件类型。例如，可以选择.a64添加汇编文件。

3 在附加过滤器输入域可以输入模式匹配字符串指定期望保留在分析器中的文件。例如，输入std\*.\*指定显示以std开头的文件。

4 在排除输入域，输入模式匹配字符串指定期望排除在分析器之外的文件，如输入temp\*.\*可以将所有以temp开头的文件排除在外。

5 勾选Include Subdirectories 复选框（默认选中）可以将选中目录的子目录添加到工程配置，这种情况下指定目录下所有满足过滤设置的子目录和文件都会添加到工程。

6 选中Watch this directory复选框可以将指定目录的文件操作动作加入监视范围，当该目录发生文件添加或者删除，Understand会向用户反映出这种变化。 表示当前目录为监视状态，而 表示当前状态为非监视状态。默认情况下一个监视目录的所有子目录都会被自动设置成监视状态，这种设置状态的修改可以参见。

7 对于UNIX系统的符号链接文件，可以选择在添加文件时是否维持其链接对象。

8 对所有选项进行设置后，点击OK将所涉及的源文件加入当前工程，添加过程的时间视操作规模而定，点击Cancel可以取消当前的添加操作。


#### 小技巧：



Understand工程可以添加多个目录树的文件。

对于Windows操作系统，拖曳操作也能够向工程配置对话框添加目录，文件或者一系列选中文件。如果拖曳对象是一个目录，一个工程目录对话框会被自动打开。如果拖曳对象是一个文件，不管是否匹配过滤条件设置，该文件会被添加到工程。

所有目录都应是绝对路径。

## 添加文件

点击可以添加指定的单个文件。点击该按钮，弹出一个文件选择对话框，可以在其中选择一个或多个文件，点击 Open，即可将选中文件添加到当前工程。

点击旁边的下拉箭头，可以选择将一个编译器应用或者代码管理系统或者其他方法生成的包含文件列表的 text 文件导入工程。Understand 要求 text 文件按照行列出每个目录的绝对路径。详见向工程添加多个文件。

## 删除目录或文件


选择期望删除的目录或文件，点击，即可从工程中删除指定对象。

## 配置重设

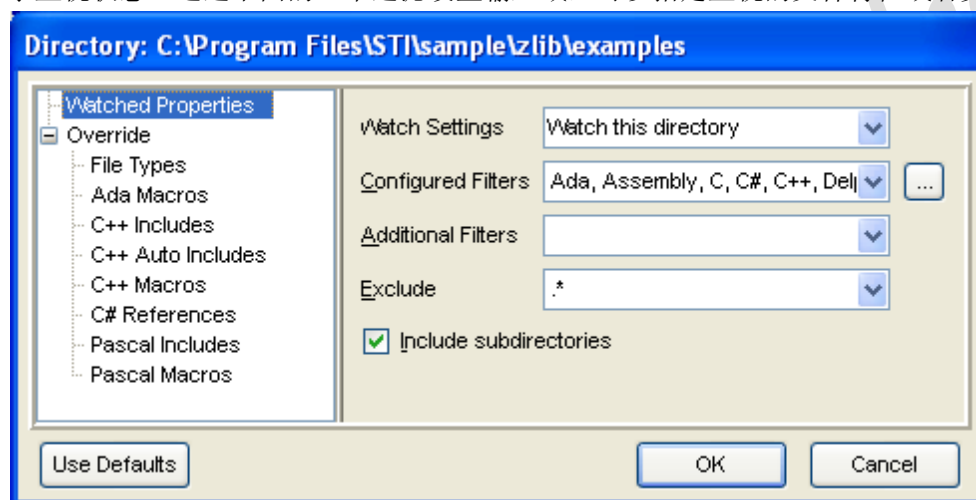
默认情况下，Understand 2.5 按照工程配置对话框中指定的按照语言类别配置的处理方式对工程中的所有文件进行处理。例如，在工程配置对话框对于所有 C++ 文件指定了包含路径和宏定义，然而后续针对单个目录甚至文件对这些设置进行修改也被 Understand 所支持。

**目录：**期望对目录的配置进行重设，可以按照如下步骤：

1 选中一个目录。

2 点击 ，或者在右键菜单中选择 Configure override settings。

3 弹出对话框中选择 Watched Properties，可以设置当该目录发生添加或删除时，目录下文件被监视的方式。在 Watch Settings 输入域，可以选择监视，不监视或者从上级目录继承监视状态。通过下面的三个过滤设置输入域，可以指定监视的文件特征或者类型。

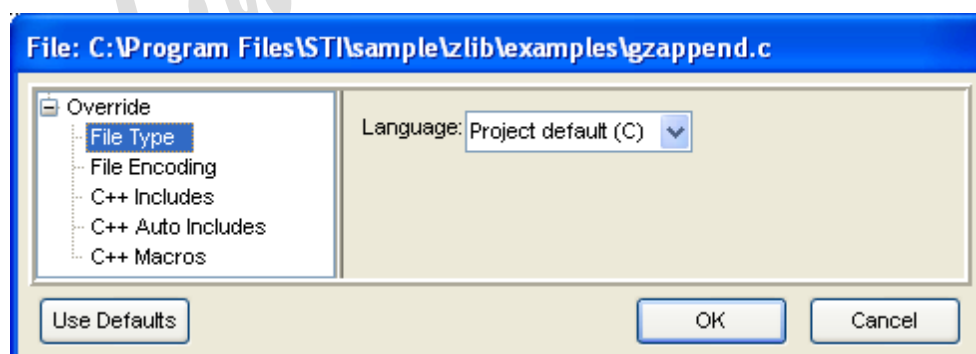


4 在 Override 下的多种分类，可以指定语言相关的一些配置，这个分类列表依赖于工程选定的语言类型。

**文件：**期望对文件的配置进行重设，可以按照如下步骤：




1 选中一个文件。

2 点击 ，或者在右键菜单中选择 Configure override settings。



3 在 Override 下的多种分类，可以指定语言相关的一些配置。分类列表依赖于选中文件的语言类型，详见。文件的配置重设只有在选择了 **Relative or Named Root portability** 才会包含 Watched Properties 分类。

4 点击 OK 保存重设内容。

在目录树结构上  表示目录的监视状态， 表示目录的配置重设状态， 表示两种状态均有。

Override 分类具有 Ignore Parent Overrides 的选项，选中情况下上级目录的设置被忽略，而只保留当前目录或者文件的重设内容。


## 扫描监视目录

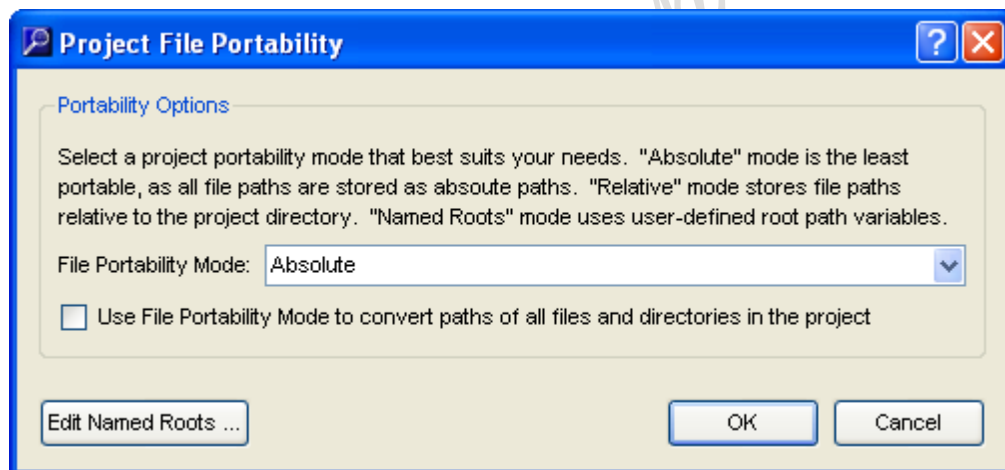
通过 **Project>Rescan Project Directories** 扫描所有被设置成监视状态的目录，检查文件添加和删除动作。

如果发现有不希望包含在工程中的文件，取消该文件旁边的复选框即可。

Understand 2.5 支持定期扫描，详见计划动作。

## 配置文件可移植性

工程配置对话框的文件类别配置页面上方的  图标可以控制 Understand 2.5 工程的文件可移植性，点击该按钮，弹出如下对话框。



工程的文件可移植性可以保证与其他用户共享工程，移动文件而不影响工程的使用。

这些选项包括：

- **Absolute:** 默认选项，工程保存所有目录的绝对路径，如果文件发生移动，原来的路径变得无效。

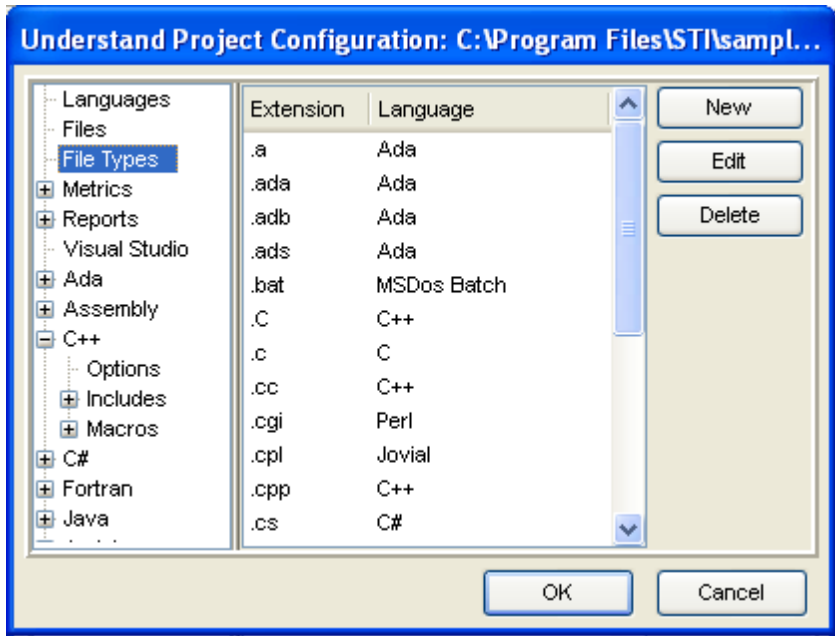
- **Relative:** 这个选项使得工程保存文件基于 Understand 2.5 工程数据库文件路径的相对路径。将工程数据库与代码保存在同一目录，移动这个目录将不会影响工程的使用。

- **Named Root:** 这个选项使得用户可以通过设置类似环境变量的“命名根”来指定一个根目录。不同的用户可以独立指定自己的“命名根”，点击 **Edit Name Roots** 按钮，详见。

勾选 **Use File Portability Mode to covert paths** 复选框可以修改当前文件路径保存方式。

# 文件类型

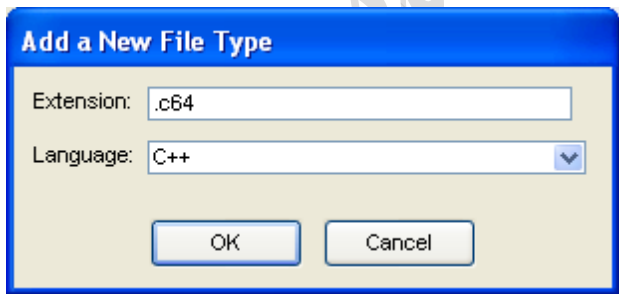
在工程配置对话框的文件类型类别配置页面，可以配置 Understand 2.5 如何识别具有不同扩展名的各种文件。



右侧列表显示了当前已经支持识别的文件扩展名，具有这些文件扩展名的文件被 Understand 解析，而其他文件则被忽略。

选择一个类型点击 **Edit** 可以修改既有类型。

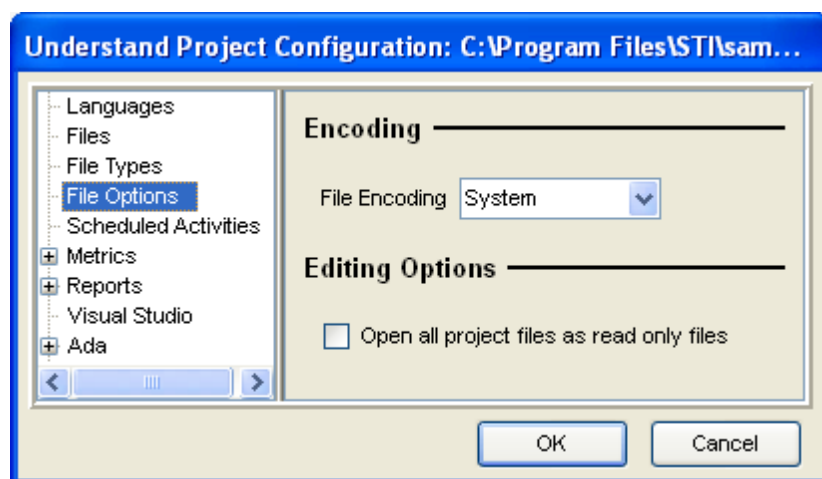
点击 **New**，可以添加一个类型，输入文件扩展名，为其指定对应的语言类型，然后点击 **OK**。



# 文件类型

工程配置对话框的文件类型配置页面，可以设置文件打开和保存方式。





- **File Encoding:** 设置保存源文件的编码方式，Understand 2.5 提供了多种编码方式。推荐仅在其他应用打开或者显示 Understand 2.5 保存的文件出现问题时使用这个设置，详见编辑器类别配置。默认编码配置为“system”，表示使用电脑使用的编码方式。修改这个设置会影响之后新建的 Understand 工程。这个设置可以针对目录或者文件进行重配置，详见配置重设。

- **Open all project files as read only files:** 不希望 Understand 2.5 对打开文件进行修改可以使用这个选项。

## 计划活动

工程配置对话框的计划活动配置页面，可以设置指定的常规活动。

**Tools>Scheduler>Scheduled Activities--<project\_name>**也可以快速打开这个页面。

对当前工程配置计划活动，可以通过如下步骤：

- 1 勾选 Process At 复选框。

- 2 选择执行时间和日期。

- 3 选择期望执行的活动，活动执行顺序按照图中所列。例如，扫描目录在分析文件之前执行，分析文件在度量处理之前执行。

**注意：** Understand 2.5 必须在计划活动指定的时间处于运行状态，否则该活动不会被执行。

以下活动可以被列为计划活动：

**Rescan watched directories:** 指定自动检查工程监视目录发生的文件添加或删除，指定监视目录的方法详见添加目录。如果存在监视目录，执行分析文件之前都应该执行该项活动。通过 **Project>Rescan Project Directories**，可以在其他时间执行该项活动。

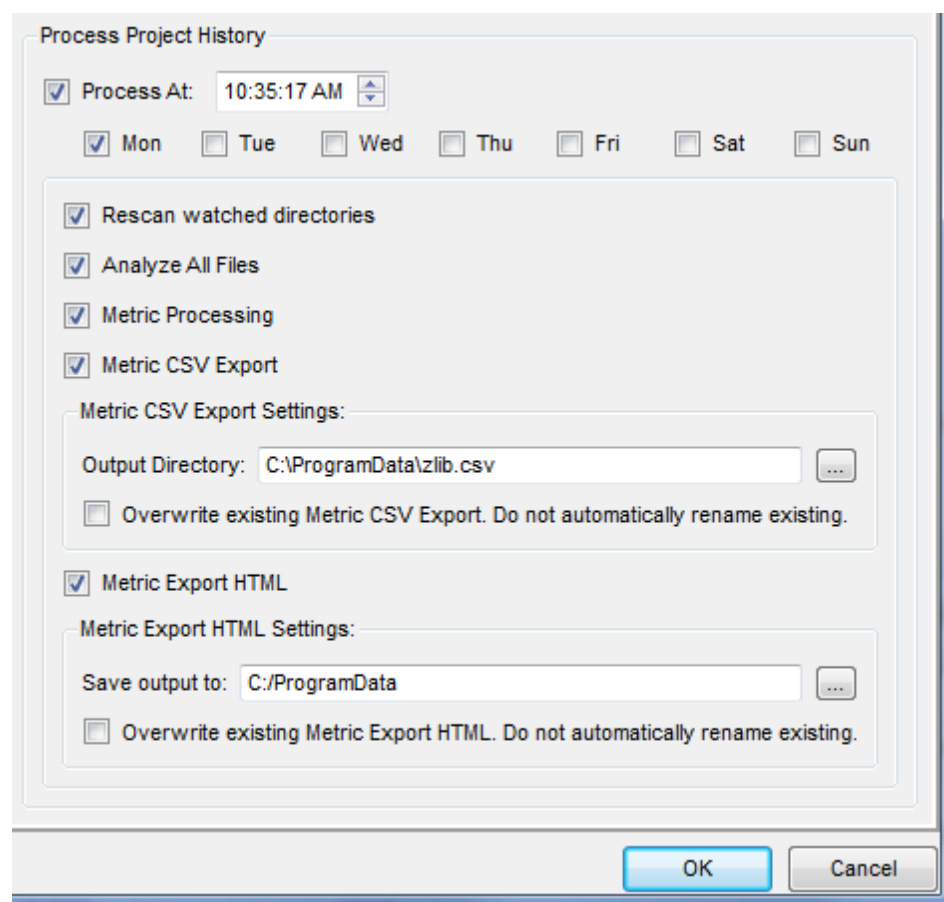
**Analyze all files:** 指定自动执行如分析代码中描述的所有文件的分析活动。确保在度量数据生成之前执行该活动以便度量结果能够实际反映工程的当前状况。通过 **Project>Analyze All Files**，可以在其他时间执行该项活动。

**Metric processing:** 指定自动计算工程的度量数据，在 **Metrics>Selected Category** 选中的度量项目会被执行。如果希望执行下面两种度量数据导出任务的任何一种，都需要将此选项勾选。

**Metric CSV export:** 指定自动将度量数据以逗号分隔的值文件导出。选中此选项的情况下，需要指定导出文件的文件名及其放置的路径。默认情况下，具有相同文件名的导出文件会被自动重命名以保留备份，如果期望替代原导出文件，可以选择 **Overwrite** 多选框。导出配置详见度量。使用 **Metrics>Export Metrics** 可以在计划活动在外执行此项活动，详见导出

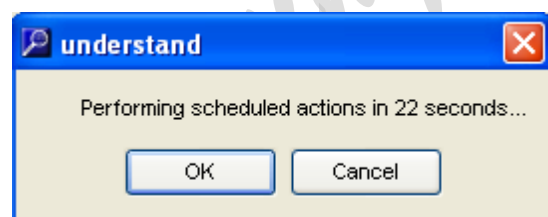


度量的 CSV 文件。



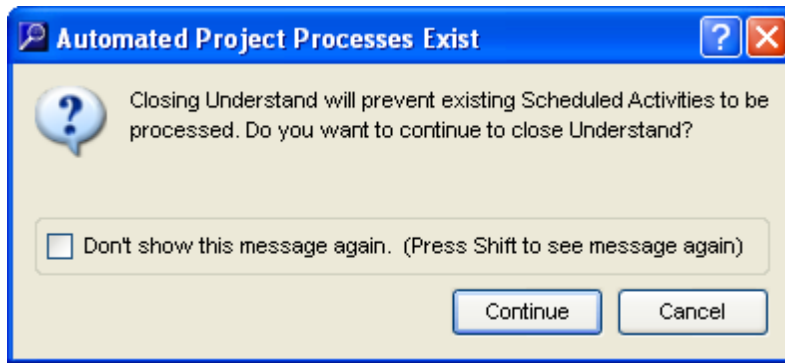
**Metrics export HTML:** 指定自动导出 web 页面格式的度量数据。勾选此选项后，指定导出文件的保存路径。默认情况下，具有相同文件名的导出文件会被自动重命名以保留备份，如果期望替代原导出文件，可以选择 **Overwrite** 多选框。使用 **Metrics>Project Reports** 可以在计划活动在外执行此项活动，详见导出度量的 HTML 文件。

计划活动开始启动之前，可以看到如下提示对话框，此时可以通过点击 **Cancel** 取消活动执行。



如果配置过计划活动，退出 Understand 2.5 时，会弹出如下提示框。

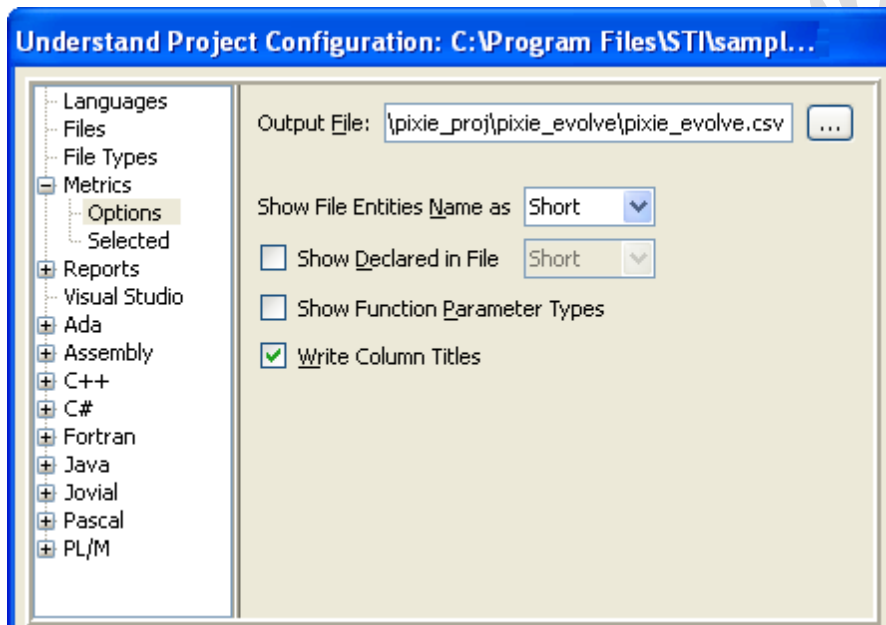
使用 **Tools>Scheduler>Scheduled Activities—All Projects**，可以检查当前已配置的计划活动。如果期望修改计划活动的执行时间，必须重新打开工程，修改工程配置对话框中的对应内容。



## 度量

在工程配置对话框的度量配置页面，可以配置导出 CSV 文件时如何生成度量数据。手动更新和计划活动更新的度量数据计算的默认选项都通过其中的选项进行配置。

度量配置包含两个子配置页面：Option 和 Selected。



通过 Project>Configure Project 打开工程配置页面，然后在左侧选择 Metrics 即可打开如上页面。如果在度量配置之前尝试生成度量数据，如上页面会自动打开。

Option 子配置页面包括如下内容：

**Output file:** 为度量输出指定保存路径和文件名。Understand 2.5 将度量数据保存到一个以逗号为分隔符的 CSV 文件，这个文件可以使用 Microsoft 的 Excel 和其他表格应用程序打开。

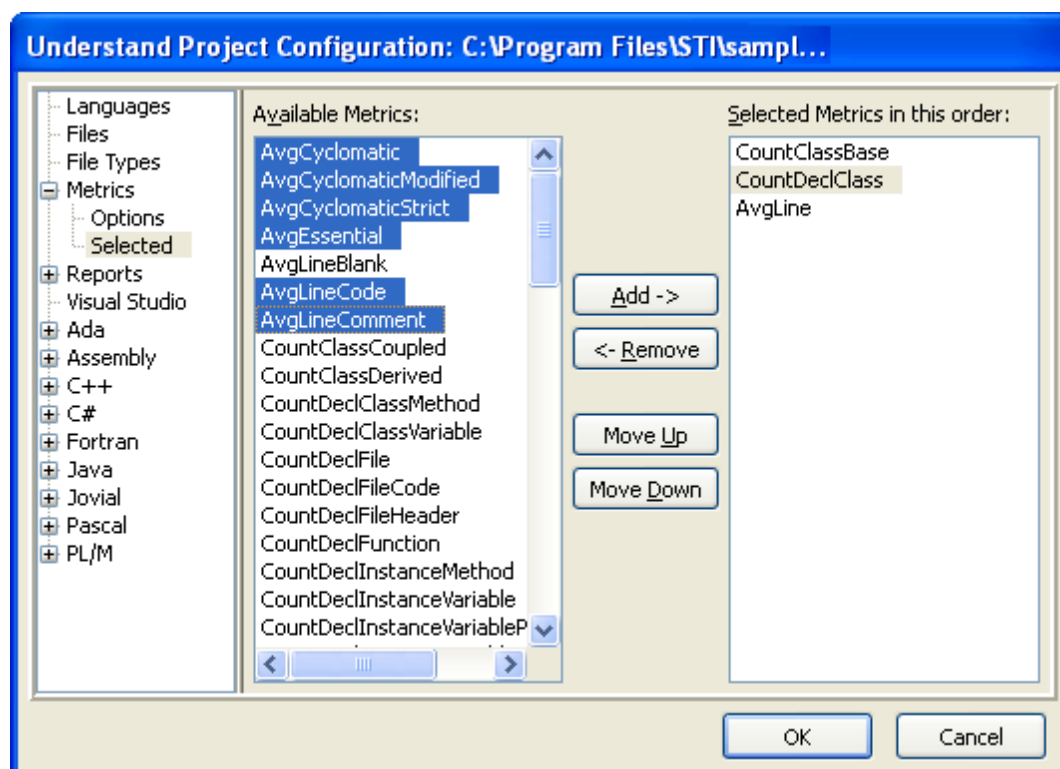
**Show File Entities Name as:** 选择文件以纯文件名（仅包含文件名），全文件名（包含绝对路径）还是相对文件名（包含相对路径）的形式显示。

**Show Declared in File:** 勾选此选项可以将包含实体声明的文件包含到输出文件，此处也可以选择将相应文件以纯文件名（仅包含文件名），全文件名（包含绝对路径）还是相对文件名（包含相对路径）的形式显示

**Show Function Parameter Types:** 勾选此选项可以将函数参数的类型列出。

**Write Column Titles:** 勾选此选项可以指定 CSV 文件以列开头。

Selected 子配置页面如下图所示。



- 1 在左侧的 Available Metrics 列表，选择期望在输出文件包含的对应项，可以使用 shift 选择多个连续项，或者使用 ctrl 选择多个不连续项。
  - 2 点击 Add，将选中内容拷贝到右侧列表。
  - 3 通过 Move up 和 Move down 对右侧列表内容进行重新排序。
- 度量数据说明可以参考 [www.scitools.com/documents/metrics.php](http://www.scitools.com/documents/metrics.php)。

## 报告

在工程配置对话框的报告配置页面，可以配置报告的生成。报告配置包括以下三个子配置内容：Output，Options 和 Selected。

选择 Project>Configure Project，点击 Report 类别可以打开如下页面，在工程报告窗口点击 Configure 也可以打开该页面。

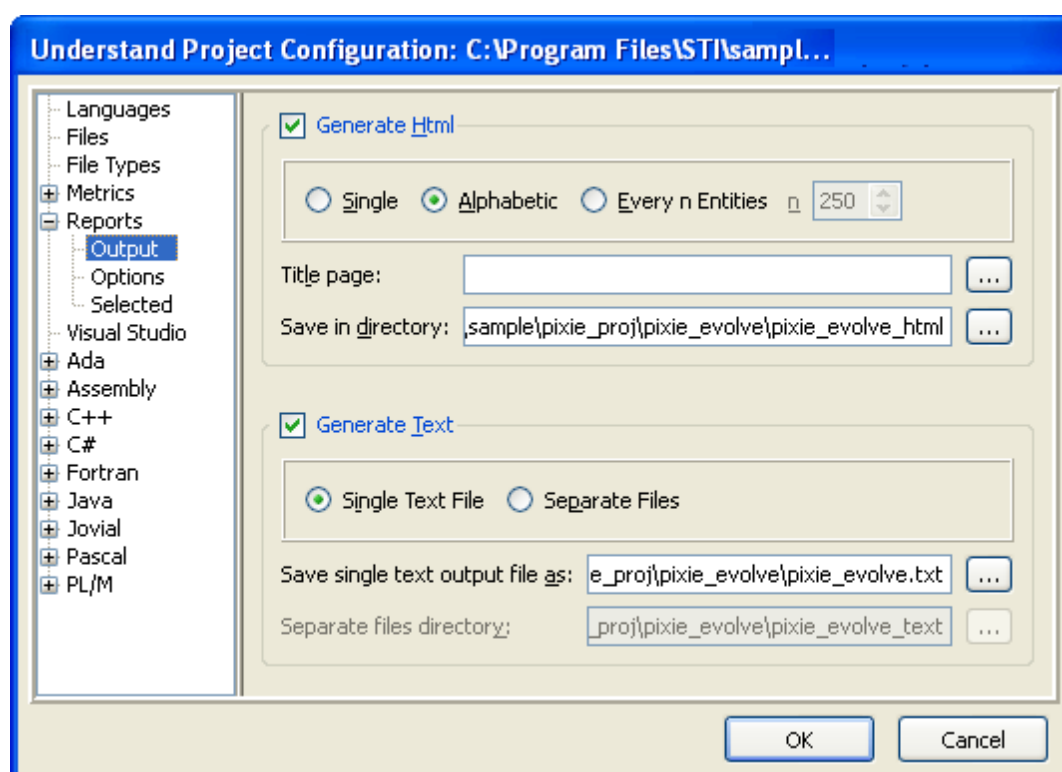
HTML 报告的颜色和字体也可以进行配置，详见定制报告颜色。

## Report>Output 子配置页面

Output 子配置页面包含两个主要区域：

- **Generate HTML:** 此选项控制生成一系列相互链接的 HTML 文件。
  - 对于每种报告类型都可以选择生成单个或者系列 HTML 文件，对于大型工程推荐生成多个 HTML 文件。选择 Alphabetic 可以选择生成以实体名称首字母排序的系列 HTML 文件，选择 Every n Entities 生成多个 HTML 文件，每个文件包含 n 个实体。默认情况下，Understand 对应每个字母生成一个 HTML 文件。
  - 报告主页命名为 index.html，用户可以按照自己喜好另外通过 Title page 指定主页。
  - Save in directory 指定报告文件的保存路径，其默认值为工程数据库文件.udb 所在

目录的<proj\_file>\_html 子目录，用户可以根据喜好进行修改。



**Generate Text:** 此选项指定生成包含报告数据的文本文件。

- 选择 **Single Text File** 生成一个指定路径和文件名的文本文件，选择 **Separate Files** 在指定路径下生成多个文本文件。文本文件的文件名指示单个报告类型。根据选项不同，可以选择输出的文件名或者保存路径。

HTML 格式和文本格式报告的生成可以独立选择。

## Report>Option 子配置页面

Option 子配置页面包括以下域：

- **Display full filenames:** 此选项控制符号树和度量报告使用全文件名显示，默认使用纯文件名。
- **Write generation time on report:** 此选项将报告生成的日期和时间包含到文本格式报告的开头，默认选中。

## Report>Selected 子配置页面

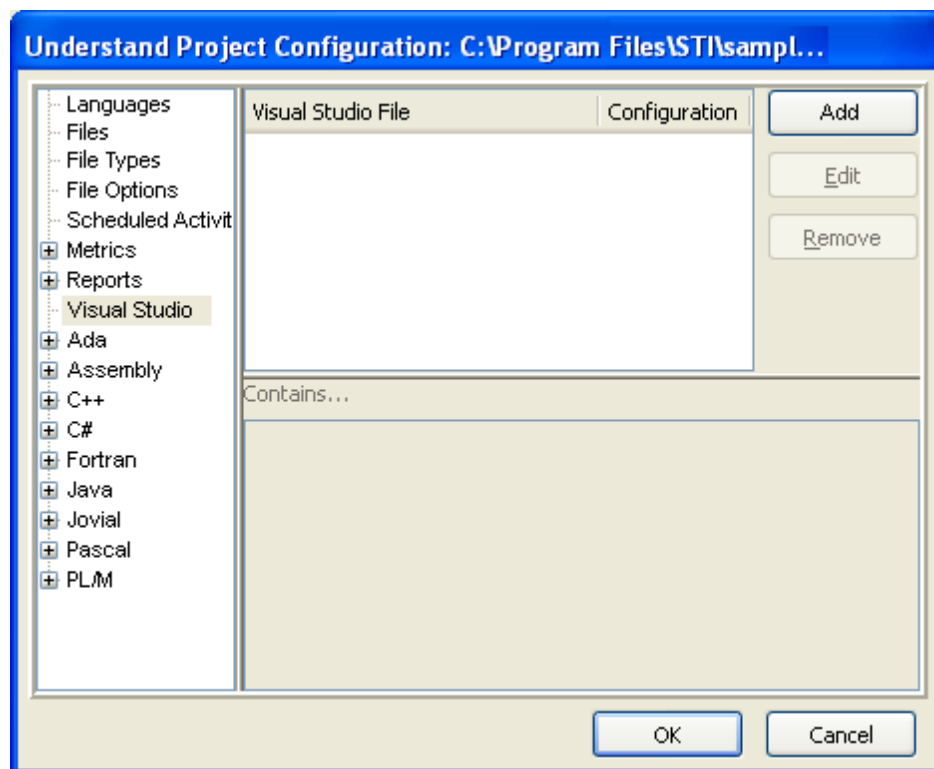
Selected 子配置页面选择生成报告需要包含的内容。报告列表内容根据工程选择的语言而定，其格式参见第 8 章。

## Visual Studio

在工程配置对话框的 Visual Studio 配置页面，可以指示 Understand 2.5 直接使用 Microsoft

Studio 工程文件中配置的源文件，宏定义以及头文件包含路径。

使用 Project>Configure Project，选择 Visual Studio 打开如下页面。



按照如下步骤：

- 1 点击 Add；
- 2 在 Add a new Visual Studio 对话框，点击...按钮进入 Visual Studio File 页面，然后指定一个 Visual Studio 工程文件，点击 Open。Understand 2.5 支持 MS Visual Studio 工程文件包括以.csproj（C#工程），.dsp，.dsw（工作空间文件），.sln，.vcp（Windows CE 工程），.vcproj（Visual C 工程），.vfproj（Visual FORTRAN 工程）以及.vcw（工作台文件）。
- 3 在 Configuration 指定期望 Understand 2.5 使用的配置项。
- 4 使用 Exclude Filter 指定导入 Visual Studio 工程时需要过滤的文件扩展名。
- 5 展开 Unfiltered Contents 列表可以检查当前已经选择的包含路径，宏定义以及文件。
- 6 点击 OK，将以上配置添加到工程。

**注意：**由于 Visual Studio 工作空间文件没有提供在.dsw 文件中为每个.dsp 工程指定目标的机制，Understand 2.5 的同步对象是 Visual Studio 工作空间文件时，只能使用其默认目标。

完成设置后，Understand 2.5 就能够使用 Visual Studio 工程中配置的源文件，宏定义及包含路径。这种配置方法与工程其他类别的配置互不冲突。

**注意：**

其他类别配置中设置的包含路径和宏定义优先于 Visual Studio 工程配置，因此用户可以在批量使用 Visual Studio 现成工程配置的情况下，对一些期望修改配置进行修改。



**Ada 选项**

在工程配置对话框的 Ada 选项页面，对 Undersand 2.5 Ada 源文件分析进行配置。使用菜单栏 Project>Configure Project，选择 Ada 可以打开如下窗口。

此配置页面包括如下区域。

- **Version:** 选择工程使用的Ada版本，支持Ada83，Ada95和Ada05。
- **Preprocessor:** 选择Ada使用的预处理声明类型，支持None，C和Verdix。
- **Standard:** 指定工程使用的标准库所在路径，默认为<install direcotry>/conf/

understand/ada。

有时候，代码编译环境比 Ada 语言参考手册中定义的“Standard”环境更有助于代码分析，尤其是当编译器提供与其他语言的兼容或者适应芯片/系统的底层特性。为了达到这个目的，将所有源代码包括 Ada 新标准定义放到一个目录，并在 Standard 域指向这个目录。

- **Case of externally linkable entities:** 指定Ada语言导出实体供其他语言链接（如函数调用）时使用的大小写。例如，源文件中声明了一个名为“MYITEM”的实体，而在此处选择了“all lowercase”，其他语言在使用该实体时，应该使用名称“myitem”。
- **Count and/or operators in strict complexity:** 勾选此选项，understand在计算精确复杂度度量时将“and”和“or”也计算在内，精确复杂度度量详见程序单元复杂度报告。精确复杂度类似圈复杂度，不同的是如“and then”和“or else”的短环路运算符也会导致该复杂度加1。
- **Count exception handlers in complexity:** 勾选此选项（默认选中），understand在计算复杂度度量（详见信息浏览器和程序单元复杂度报告）将异常处理逻辑也考虑在内。
- **Count for-loops in complexity:** 取消此选项，understand在计算复杂度度量（详见信息浏览器和程序单元复杂度报告）时不考虑FOR循环，仅仅计算程序单元独立路径数目。

---

**Compiler**

Version Ada95  
Preprocessor None  
Standard C:\Program Files\STI\conf\understand\ada\ada95\ ... Reset

**Multiple Language Linkage**

The case of externally linkable entities is  
☒ all lowercase ☐ all uppercase

**Metrics**

☐ Count and/or operators in strict complexity  
☒ Count exception handlers in complexity  
☒ Count for-loops in complexity

**Optimize**

☐ Create and cross reference record object components  
☐ Create relations between formal and actual parameters  
☐ Less memory usage versus speed  
☒ Save comments associated with entities

**Options**

☒ Prompt on parse errors  
Display entity names as Original  
Main subprograms: (main1, main2..)

- **Create and cross-reference record object component:** 勾选此选项（默认取消），一个记录类型的所有参数和对象被创建成独立实体，默认情况下，对象部件的所有引用被当做记录类型部件的引用。

- **Create relations between formal and actual parameter:** 勾选此选项，形参与实参之间建立关联关系，关联到形参的实参包含作为实参传递的表达式中使用的条目。为了加速分析过程，此选项默认取消。

- **Less memory usage versus speed:** 勾选此选项，Understand 2.5 消耗极小量的内存，这种模式下，Understand 在程序单元不再需要时，释放其使用的内存，因此会显著影响运行速度。默认情况下，此选项取消选中。

- **Save comments associated with entities:** 此选项控制代码实体前后注释与实体的关联。

- **Prompt on parse errors:** 默认情况下，文件分析过程发生错误时会提示如何进行后续处理，用户可以选择忽略该错误以及后续所有错误。禁止该选项可以取消弹出窗口。

- **Display entity name as:** 选择 Understand 2.5 显示的实体名称大小写方式，包括按照源代码显示，全大写显示，全小写显示，首字母大写显示或者混合显示。

- **Main Subprograms:** 以逗号分割列表显示工程中主要子项目名称。



## Ada>Macro 子配置页面

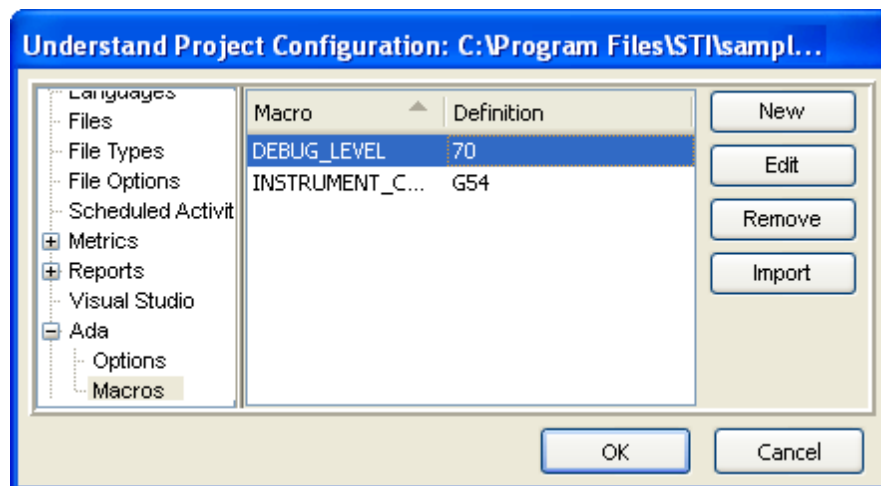
Ada 语言通过预处理语句指定条件编译，如

```
PRAGMA IF DEVICE == D129
```

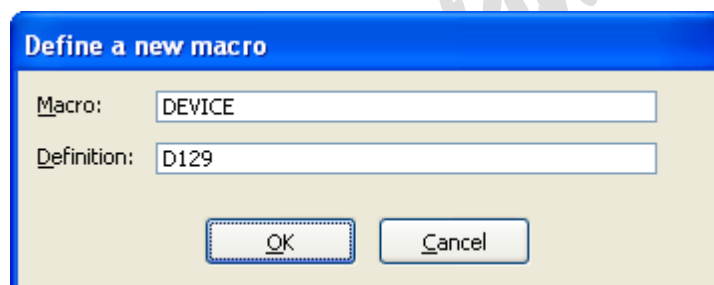
支持的语句包括 IF, IFDEF, ELSIF, ELSE 和 ENDIF，这些语句功能与 C 语言的预处理语句如 #ifdef 类似。

为了正确分析代码，Understand 2.5 需要知道哪些宏定义被设置过。

在工程配置对话框的 Ada>Macro 配置页面，可以设置源代码中相关联的宏定义。菜单栏选择 Project>Configure Project，点击 Ada 配置页面的 Macro 分类，可以看到如下窗口。



该页面显示了宏定义名称及其具体定义，每个宏定义条目可以被删除或者编辑。点击 New 新增一个宏定义。



在对应域输入宏名称和定义值，然后点击 OK。

宏名称为必须项，而定义值为可选项。没有定义值的宏普遍被预处理语言 IFDEF 用来确认某个宏是否被定义过。

选择一个宏定义，点击 Edit 可以在不修改宏名称的前提下，对其定义值进行重新设置。

通过 Import，选择一个文本文件，可以导入其中的宏定义及其定义值，对象文件需要每行包含一个宏定义，行首的“#”表示该行为注释行，使用“=”分隔宏名称和宏定义值，如 DEBUG=true。

通过 und 命令行使用 -define name[=value] 也可以设置一个宏定义。

## C++选项

在工程配置对话框的 C++选项页面，对 Undersand 2.5 C/C++源文件分析进行配置。使用菜单栏 Project>Configure Project，选择 C++可以打开如下窗口。



**Compiler**

Compiler: Microsoft Visual C++

Compiler include paths: %include%

☐ Allow nested comments

**Multiple Language Linkage**

Prepend the names of externally linkable entities with:

Append the names of externally linkable entities with:

**Optimize**

- ☐ Create implicit special member functions
- ☒ Create references in inactive code
- ☒ Create references to local objects
- ☐ Create references to macros during macro expansion
- ☒ Create references to parameters
- ☒ Save comments associated with entities
- ☐ Save duplicate references
- ☐ Save macro expansion text
- ☒ Use include cache

C++>Option 配置页面包括如下区域:

- **Compiler:** 选择当前工程使用的编译器或者平台, 编译器相关的宏定义根据选择自动设置。需要注意的是, 编译器的某些特性有可能没有得到很好的支持。
- **Compiler Include Paths:** 输入编译器使用的头文件的所在路径, 如%include%。
- **Allow nested comments:** 默认情况下, 该选项禁止。打开的状态下, Understand 允许 C 语言的注释符 (/\*\*/) 可以嵌套使用 (这种风格被 ANSI 禁止, 但是有一些编译器支持)。
- **Prepend the names of externally linkable entities with:** 使用一个随意的字符串作为工程中其他语言编写的代码中定义的实体的前缀。
- **Append the names of externally linkable entities with:** 使用一个随意的字符串作为工程中其他语言编写的代码中定义的实体的后缀。
- **Create implicit special member functions:** 源代码没有类和结构实体的相关语句时, Understand 数据库自动创建默认构造函数, 析构函数和隐式声明引用, 从而为分析时提供相关的引用实体。该选项默认禁止。
- **Create references in inactive code:** 如果希望将条件编译控制的非激活代码排除在外, 需要取消该选项, 默认选中。
- **Create references to local objects:** 默认情况下, Understand 数据库包含所有局部对象, 如果希望不包含函数中声明的变量需要取消该选项。Understand 2.5 的主窗口可以选择是否需要在 HTML 报告中包含局部对象。
- **Create references to macros during macro expansion:** 选中情况下, 数据库保存宏解析时的引用关系。有时候, 该选择有用。注意, 该选项选中导致数据库增加很多引用关系, 会变得很慢。默认关闭。
- **Create references to parameters:** 关闭该选项取消参数的引用关系, 默认开启。
- **Save comments associated with entities:** 此选项控制代码实体前后注释与实体的关联。
- **Save duplicate references:** 默认情况下, 引用关系只在数据库保存一份, 选中该选

项，会记录重复的引用关系。

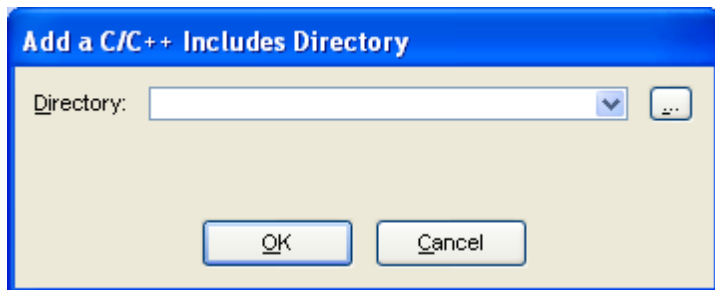
- **Save macro expansion text:** 选中该选项的情况下，可以在选中宏的右键菜单下选择 Expanded Macro Text 来查看宏定义值。
- **Use Include cache:** 由于头文件通常被多个源文件使用，默认情况下，分析阶段头文件都采用高速缓存，这样能够加速分析过程，但是需要更多的内存，如果分析过程碰到内存问题，可以将此选项关闭。同时需要注意的是，高速缓存的使用会影响到分析结果。

## C++>Include 子配置页面

工程配置对话框的 C++>Include 子配置页面对包含路径进行设置，一个工程可以指定多个包含路径。

包含路径不支持递归查找，也就是说，Understand 不会查找没有在包含路径列表中指定的子目录。

点击 New，然后...选择期望的路径，点击 OK 就可以新增一个包含路径。



分析过程，Understand 按照该对话框的顺序查找各个目录，通过 Move Up/Move Down 来调整目录的查找顺序。通常情况下，这里只需要包含与当前工程不直接相关（如系统头文件）或者不期望 Understand 对其进行完整分析的头文件。对于期望对其进行分析的工程文件应该通过 Source 配置将其作为源文件加入工程。

在指定文件路径时可以使用环境变量或者命名根目录（详见 [Named Root](#)），UNIX 环境变量引用使用 \$var，Windows 环境变量使用 %var%。

通过 Import 可以导入一个指定的包含目录列表的文本文件，该文件格式要求每个路径占用一行（行首“#”表注释），支持绝对路径和相对路径（相对路径以工程文件所在目录为基准）。

C++>Include 配置页面提供如下选项控制头文件包含处理动作：

- **Add found includes files to source list:** 默认关闭，使能该选项使得工程分析时涉及的头文件自动加入到工程，从而可以获取相关文件的更多信息。
- **Add found system include files to source list:** 默认关闭，在使能前一选项的情况下，该选项可以被开启以添加相关的系统头文件。
- **Prompt for missing include files:** 默认情况下，分析过程中如果不能找到头文件会弹出提示窗口，用户可以选择向包含路径增加一个目录，忽略该文件或者禁止此类告警。关闭该选项，导致 Understand 在分析过程发现未知头文件时不进行任何提示。
- **Search for include files among projet files:** 默认开启，Understand 在找不到头文件时，会到工程文件中查找。
- **Treat system includes as user includes:** 默认开启，此选项控制分析过程像对普通头文件（使用引号包含）一样对系统头文件（使用 <> 包含）进行处理。该选项关闭的情况下，Understand 将只在编译器配置的路径中对系统头文件进行查找。

---

- **Use case-insensitive lookup for includes:** 默认开启，此选项控制 Understand 是否对 `#include` 语句中文件名的大小写进行区分。

- **Ignore directories in include names:** 默认关闭，开启该选项可以通知 Understand 忽略 `#include` 语句中的目录定义，直接使用在工程中找到同名文件。

## C++>Includes>Auto 子配置页面

通过 C++>Includes>Auto 子配置页面，用户可以指定期望在所有工程文件之前包含的头文件。

点击 **New**，选中对应文件，然后点击 **Open** 即可添加一个这样的文件。

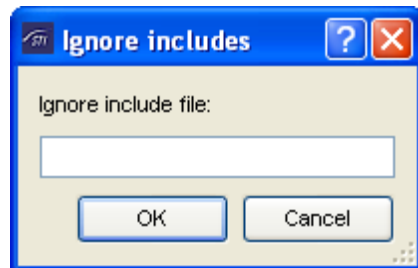
通过 **Import** 可以指定导入一个文件列表的文本文件，该文件格式要求每个路径占用一行（行首“#”表注释）。

使用 **Move Up/Move Down** 调整文件包含顺序。

## C++>Includes>Ignore 子配置页面

通过 C++>Includes>Ignore 子配置页面，用户可以指定期望不进行分析的头文件。

点击 **New**，输入期望忽略的文件名称，然后点击 **Open** 即可添加一个这样的文件，指定文件名称支持通配符，如 `moduleZ_*.h`，可以指定一系列匹配文件。



分析过程中选择忽略的未知文件也会自动加入到此类列表。

通过 **Import** 可以指定导入一个文件列表的文本文件，该文件格式要求每个路径占用一行（行首“#”表注释）。

## C++>Includes>Replacement Text 子配置页面

通过 C++>Includes>Ignore 子配置页面，用户可以指定期望对头文件文本内容进行替换的文本。例如，通过此项配置，将 VAX/VMS 包含路径如 `[sys$somewhere]` 替换成有效的 UNIX 或者 Windows 路径，而不必修改源代码。

在弹出对话框的 **Include String** 域输入实际头文件中的字符串，在 **Replace with** 域输入期望被替换的字符串，点击 **OK**，即可增加一个替换条目。

通过 **Import** 可以导入包含字符串和替换目标的文本文件，该文件每行包含一个字符串，使用等号“=”分割字符串和替换目标。

使用 **Move Up/Move Down** 可以调整各个字符串的替换顺序。

## C++>Macros 子配置页面

C语言代码通常包含许多预编译指令，用以控制编译器的执行。如下的指令能够影响到Understand的执行和分析结果：

```
#define INSTRUMENT_CODE
#ifdef INSTRUMENT_CODE
... statements ...
#endif
```

通常宏定义通过头文件中预处理指令（`#include`），或者编译命令选项（`-D`）来定义。为了使Understand 2.5能够正确分析用户代码，相关宏定义必须通知给本软件。为此Understand 2.5提供了工程配置对话框的C++>Macros配置页面。

点击 **New**，在对应域输入宏名称和定义值，然后点击 **OK** 新增一个宏定义。宏名称为必填项，而定义值为可选项。没有定义值的宏普遍被预处理指令`#ifdef`用来确认某个宏是否被定义过。

### 注意：

一些处理器相关的宏 Understand 提供了自动支持。

除了普通的宏，Understand 2.5 还支持嵌入式汇编代码使用的如下宏定义格式：

```
#asm(<embedded assembly code>);
#asm "<embedded assembly code>";
#asm<embedded assembly code>#endasm
```

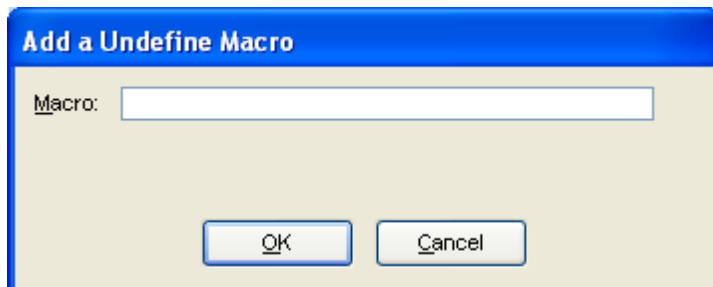
通过 **Import**，选择一个文本文件，可以导入其中的宏定义及其定义值，对象文件需要每行包含一个宏定义，行首的“`#`”表示该行为注释行，使用“`=`”分隔宏名称和宏定义值，如 `DEBUG=true`。

宏定义优先级从低到高如下所示：

- 1 语言内嵌宏（如 `__FILE__`）
- 2 编译器配置文件
- 3 同步 Visual Studio 工程定义的宏
- 4 编译单元定义的取消宏（通过 **Configure Undefines** 按钮）
- 5 工程定义的宏（**Macro子配置页面**）
- 6 `und` 命令行使用`-define`定义的宏
- 7 源文件中定义的宏（`#define/#undef`）

## C++>Macros>Undefines 子配置页面

通过工程配置对话框的C++>Macros>Undefines子配置页面可以取消宏定义。点击 **New**，输入宏名称，然后点击 **OK**。



---

通过 Import, 选择一个文本文件, 可以导入其中取消的宏定义, 对象文件需要每行包含一个宏名称, 行首的“#”表示该行为注释行。

C#选项

FORTRAN 选项

Java 选项

JOVIAL 选项

Pascal 选项

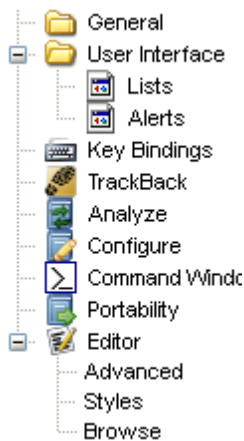
PL/M 选项

VHDL 选项

Web 选项

## 设置惯用选项

通过 Tools>Options, 打开 Understand 选项对话框, 用户可以从各个方面控制 Understand 2.5 的运行。此对话框提供的选项如下图所示。



以下章节针对每个分类进行分别描述:

- General
- User Interface
- User Interface>Lists
- User Interface>Alerts
- User Interface>Windows
- Key Bindings
- Analyze
- Configure
- Command Window
- Portability
- Dependency
- Editor
- Editor>Advanced
- Editor>Styles
- Editor>Browse

- Editor>External Editor

## General

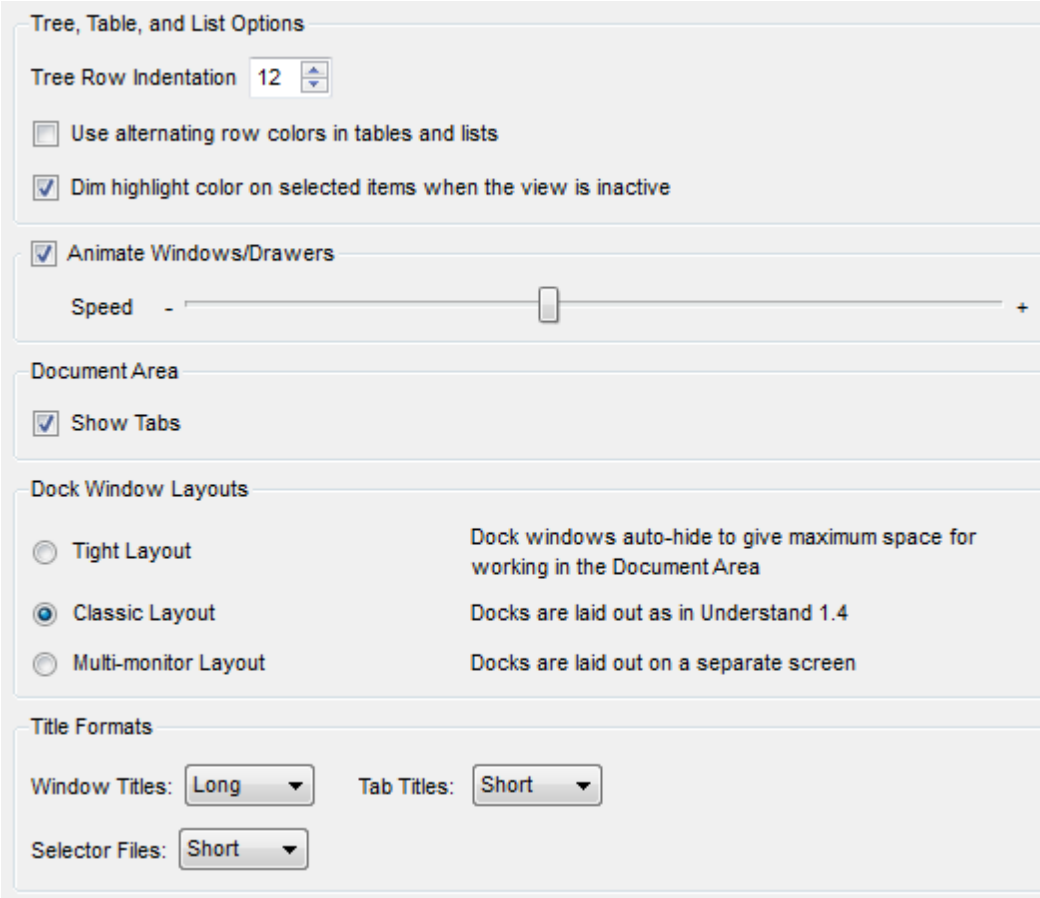
General 配置页面可以针对以下选项进行设置：

The screenshot shows the 'General' configuration window. It includes sections for 'Application Font', 'Show on Startup', 'Auto Loading/Saving Options', and 'Performance'. The 'Application Font' section shows 'Font: Arial' and 'Size: 8' with a 'Change Font ...' button. The 'Show on Startup' section has two checked checkboxes: 'Show the Splash-Screen on startup' and 'Show the Getting Started Dialog on startup'. The 'Auto Loading/Saving Options' section has three unchecked checkboxes: 'Save all modified editor windows when application loses focus.', 'Open last project on startup (currently: C:\Program Files\STI\sample\zlib\zlib.udb).', and 'Use Default Working Directory'. Below this is a text field with 'C:\Program Files\STI\sample\pixie\_proj' and a 'Browse...' button. The 'Performance' section has two checkboxes: 'Enable permissions checking for NTFS filesystems' (unchecked) and 'Allow interactivity during intensive processing' (checked). At the bottom, it says 'Allow events processing every 100 milliseconds' with a spin box set to 100.

- **Application font:** 点击 Change Font..., 选择期望使用的字体和字号, 然后点击 OK 即可设置 Understand 2.5 对话框和列表使用的字体。
- **Show the Splash-Screen on startup:** 默认开启, 启动软件时显示 logo。
- **Show the Getting Started dialog on startup:** 默认开启, 启动软件后在文档区域显示 Getting Started 标签页。
- **Save all modified editor windows when application loses focus:** 默认关闭。开启的情况下当选中其他应用程序时, understand 自动保存编辑器窗口的修改内容。
- **Open last project on startup:** 默认关闭。开启的情况下, 当不指定工程启动 Understand 2.5 时, 自动打开最近上一次使用的工程, 对于仅关注一个工程的用户尤其有用。
- **Use default working directory:** 默认关闭。开启的情况下, 用户可以选择一个自定义的默认目录, 使用该目录作为浏览目录的起始位置和相对路径的基准, 其默认值为工程保存目录。
- **Enable permissions checking for NTFS filesystems:** 开启的情况下, 在 NTFS 文件系统编辑器打开文件进行修改前 Understand 会对文件权限进行检查。由于此特性在某些场合会严重影响软件运行效率, 此选项默认关闭,
- **Allowing interactive during intensive processing:** 默认开启, 支持执行后台任务时的前台交互, 交互动作以毫秒为单位在运行间隔得到执行。
- **Reusing windows when requesting a similar style of graph from within window:** 默认开启, 在相同的窗口打开新的图片, 打开一个图片窗口列表中的项目的图片时, 此选项生效。

## User Interface

User Interface 配置页面可以针对以下选项进行设置：



The screenshot shows a configuration window with the following sections:

- Tree, Table, and List Options**
  - Tree Row Indentation: 12 (with up/down arrows)
  - ☐ Use alternating row colors in tables and lists
  - ☒ Dim highlight color on selected items when the view is inactive
- Animate Windows/Drawers**
  - ☒ Animate Windows/Drawers
  - Speed: A slider control ranging from - to +.
- Document Area**
  - ☒ Show Tabs
- Dock Window Layouts**
  - ☐ Tight Layout: Dock windows auto-hide to give maximum space for working in the Document Area
  - ☒ Classic Layout: Docks are laid out as in Understand 1.4
  - ☐ Multi-monitor Layout: Docks are laid out on a separate screen
- Title Formats**
  - Window Titles: Long (dropdown)
  - Tab Titles: Short (dropdown)
  - Selector Files: Short (dropdown)

- **Tree Row Indentation:** 控制层级树形结构显示的缩进数量。
- **Using alternating row colors in tables and lists:** 默认关闭，开启情况下表格和列表采用隔行着色显示。
- **Dim highlight color on selected items when the view is inactive:** 默认情况下在 Windows 操作系统，当一个窗口失去焦点，其中被选中目标的高亮显示被取消。如果对这种实现不适应，可以取消该选项。
- **Animate Windows/Drawers:** 默认开启，选中情况下以动画效果显示窗口和标签页的开关的动作，动画速度可以根据喜好进行调节。
- **Show tabs:** 默认开启，文档区域打开的窗口以标签页的形式在区域上方显示，支持的窗口包括源代码编辑窗口，图形窗口等。
- **Dock Window Layout:** 选择默认使用的坞窗口布局。期望尽可能多的显示空间提供给打开的多个源代码文件推荐使用 **Tight Layout**；**Classic Layout** 沿用前一版本的风格；**Multi-monitor Layout** 适合于拥有多个显示器的用户。
- **Title Format:** 用于设置窗口，标签页以及选择器文件的标题区域显示的文件名格式，包括纯文件名，全路径文件名以及相对路径文件名。



## User Interface>Lists

User Interface>Lists 配置页面可以针对以下选项进行设置：

Default Most Recently Used List Setting

Display  as the default for recently used lists.

☒ 'Recent Files' Most Recently Used List

☒ Use Default Most Recently Used Setting

☐ Display  items in the 'Recent Files' list.

☒ 'Recent Projects' Most Recently Used List

☒ Use Default Most Recently Used Setting

☐ Display  items in the 'Recent Projects' list.

- **Default most recently used list setting:** 默认情况下一个近期使用列表显示近期使用的 5 个条目，该数目可以在此进行修改。
- **Recent files most recently use list:** 选择 Display 条目，修改输入域中的数字可以修改菜单栏 File 中显示的近期使用的文件数目。取消该选项，可以省略 File 菜单栏中的近期使用文件显示。
- **Recent projects most recently use list:** 选择 Display 条目，修改输入域中的数字可以修改菜单栏 File 中显示的近期使用的工程数目。取消该选项，可以省略 File 菜单栏中的近期使用工程显示。

## User Interface>Alerts

User Interface>Alerts 配置页面可以针对以下选项进行设置：

Save On Parse

☒ Always Prompt

☐ Save modified files before parsing

☐ Don't save modified files before parsing

Save On Command

☒ Always Prompt

☐ Save modified files before running a command

☐ Don't save modified files before running a command

Project Open

☒ Prompt before closing the current project

- **Save on parse:** 选择数据库进行解析前对修改而未进行保存的文件的处理。默认的 Always prompt 选项会使 Understand 弹出窗口供用户选择是否需要保存；另外两种选项则控制



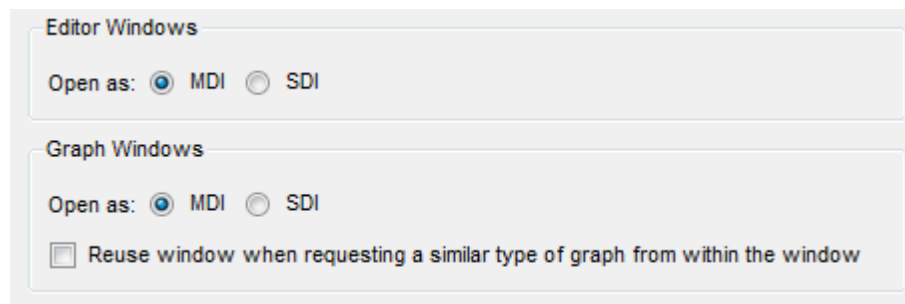
Understand 自动保存或者忽略改动。

- **Save on command:** 选择一条命令执行前对修改而未进行保存的文件的处理。默认的 Always prompt 选项会使 Understand 弹出窗口供用户选择是否需要保存；另外两种选项则控制 Understand 自动保存或者忽略改动。

- **Prompt before closing the current project:** 默认开启，打开另一个工程时，弹出窗口询问是否需要关闭当前工程及相关窗口。

## User Interface>Windows

User Interface>Windows 配置页面可以针对以下选项进行设置：

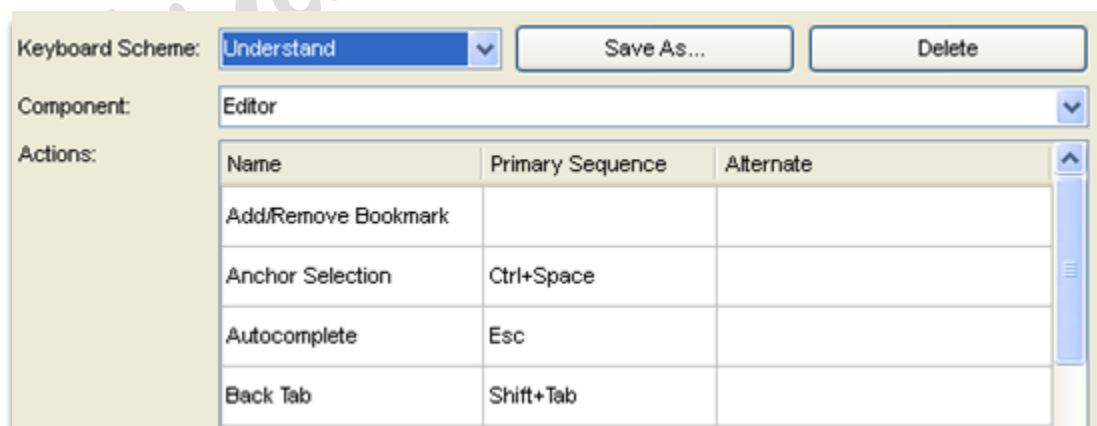


用户可以选择以多文档界面 MDI 或者单文档界面 SDI 形式打开源代码窗口和图形窗口。MDI 窗口将所有文档包含在 Understand 2.5 主窗口的文档区域，SDI 窗口针对每个文档创建独立窗口，每个窗口可以分别放在桌面上的任何位置。Understand 默认打开 MDI 窗口。

对于图形窗口，用户还可以选择在打开不同实体的同类型图形是使用同一窗口，默认选项是针对每个图形打开一个窗口。

## Key Bindings

Understand 2.5 支持对键盘功能进行定制。Key Bindings 配置页面可以对 Understand 的键盘功能进行设置。



- **Keyboard Scheme:** 本选项提供类似其他应用程序的键盘设置选择，默认选择为 Understand 自定义的键盘设置，另外选项包括 Visual studio .NET 键盘设置和 Emacs 编辑器键盘设置，选择某个方案点击 OK，即可使用选中方案。对选中的既有方案进行修改，方案即变成“定制”方案，点击 Save As...设置名称即可保存该“定制”方案。

• **Componet:** Understand 2.5 的键盘功能因操作区域不同也有差异，这些区域包括应用（对话框和列表条目），变更，比较预览，编辑器，收藏夹，文件查找，查找结果，信息浏览器，工程浏览器，替换预览和用户工具。

通过 **Help>Key Bindings** 可以查看所有键盘设置。

通过如下步骤可以修改键组合触发的动作。

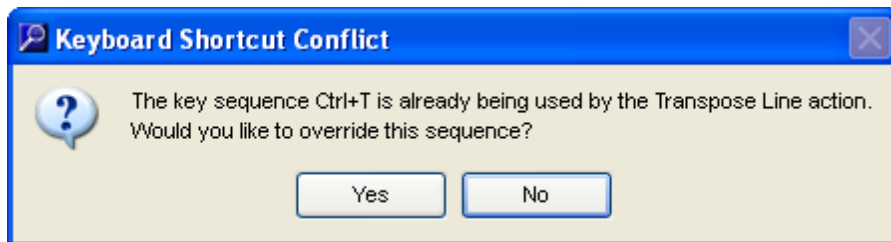
1 在 **Componet** 列表选择动作所处的区域，对于发生在多种区域的动作，可以对这些区域的动作进行统一修改。

2 在 **Action** 列表选择需要修改的动作。

3 将光标置于 **Primary Sequence** 或者 **Alternate** 列。

4 输入动作对应的键组合，注意平常的编辑键如 **Backspace** 或者 **Delete** 不能对该区域进行操作。

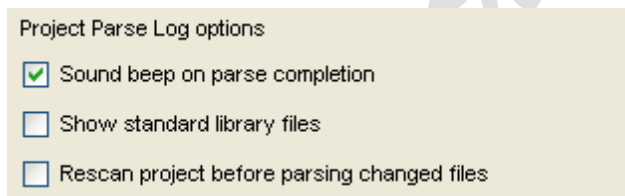
5 如果输入的键组合已经被使用，当该窗口失去焦点时，会弹出如下对话框：



6 点击 **Yes** 或 **No** 选择是否保存修改。如果不希望保存修改可以使用 **Restore Defaults** 或者 **Cancel** 按钮，选择 **Keyboard Schemes** 中的某一项也可以将键盘设置恢复成默认状态。

## Analyze

**Tools>Options** 对话框的 **Analyze** 分类对工作如何分析进行配置。



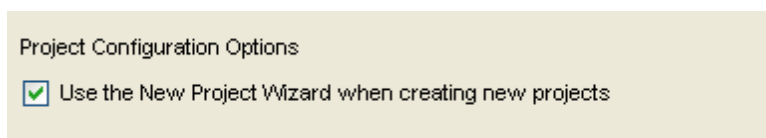
• **Sound beep on parse completion:** 默认情况下，分析完成后会发出声音提示。

• **Show standard library files:** Understand 2.5 预先解析了某些语言（如 **Ada**）标准库文件，对于这些语言，如果勾选此项解析日志会包含相关标准库文件。为了缩短日志，此选项默认取消。

• **Rescan project before parsing changed files:** 勾选此选项，Understand 2.5 在开始分析文件之前会对加入工程目录以及相关的 **Visual studio** 工程的文件进行扫描，这个动作与使用 **Project>Rescan Project Directories** 效果一样。默认情况下，此选项取消。

## Configure

**Tools>Options** 对话框的 **Configure** 分类对如下选项进行配置。

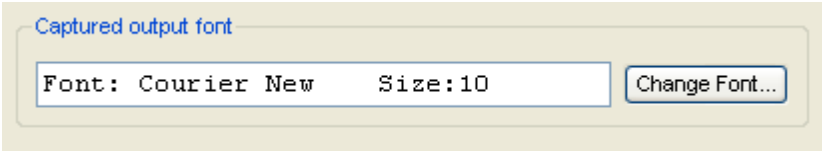


• **Use the New Project Wizard when creating new projects:** 选择此选项，点击

**File>New>Project** 时会弹出新工程引导程序。如果取消此选项，指定工程数据库路径和文件名后使工程配置对话框进行完整配置。

**Command Window**

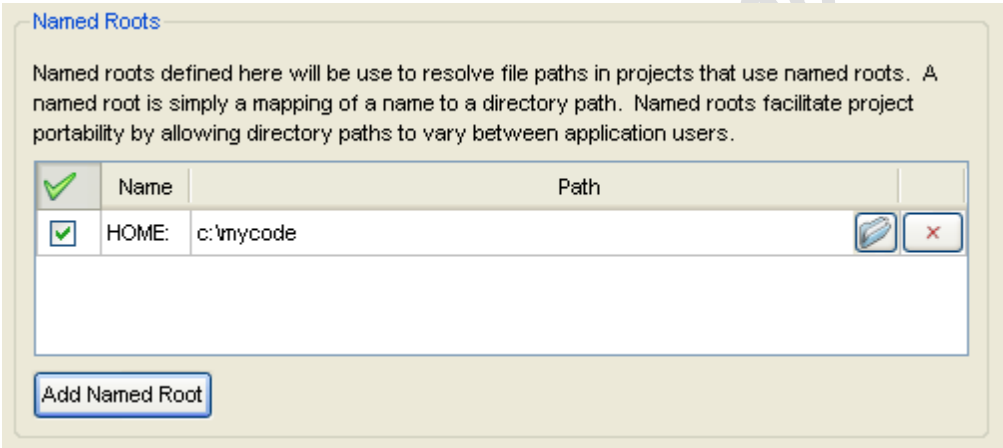
Tools>Options 对话框的 Command Window 分类对如下选项进行配置。



此选项对命令行输出对话框中字体进行设置。

**Portability**

Tools>Options 对话框的 Portability 分类用于设置指定路径的名称。命名根与环境变量类似。



指定一个命名根以后，该名称可用于 Understand 2.5 的其他窗口，如工程配置对话框，“und”命令行等。通过网络共享当前工程的其他用户可以方便地通过命名根引用工程中的文件。

点击 **Add Named Root**，在新增的行中输入名称和路径（或者点击文件夹图标选择）即可添加命名根。

通过控制命名根行前的多选框可以临时取消指定的命名根。

大多数情况下，当命名根发生变化时，工程需要进行重新分析。

Understand 2.5 可以使用指定的操作系统环境变量作为命名根。在操作系统层面，指定环境变量使用“UND\_NAMED\_ROOT\_”做前缀。在 Understand 2.5 使用命名根时不需包含其前缀。例如，指定一个系统环境变量如下：

UND\_NAME\_ROOT\_SOURCEDIR=c:\my\project\dir

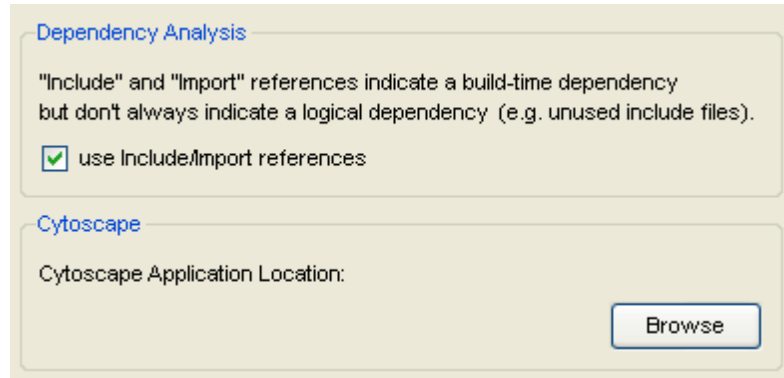
则在 Understand 2.5 中使用的命名根为 SOURCEDIR。

命名根的使用，参见[配置文件可移植性](#)。

---

## Dependency

Tools>Options 对话框的 Dependency 分类用于对依赖浏览器，依赖视图以及依赖关系导出的相关选项进行设置。



- **Use Include/Import Reference:** 默认情况下，“include”和“import”都被看做是依赖关系。然而，如果不是逻辑依赖而只是构建需要，用户可以选择在依赖列表中忽略这种依赖。
- **Cytoscape Application Location:** Cytoscape 是一种专门用于可视化分析的开源软件。设置 Cytoscape 的安装位置，Understand 2.5 可以直接使用 Cytoscape 对工程导出的包含依赖关系的 XML 文件进行查看，导出参见导出依赖度量。

## Editor

Tools>Options 对话框的 Editor 分类用于对以下选项进行设置：

The screenshot shows the 'Preferences' dialog box for Understand 2.5, organized into several sections:

- Default Style:** Font: Courier New, Size: 10, ☒ Antialias.
- File Mode:** Encoding: System, Line Endings: CRLF (Windows).  
**On Save:** ☐ Convert existing line endings, ☐ Convert tabs to spaces, ☒ Add newline at end of file if absent, ☐ Remove trailing whitespace.
- Caret Line:** ☒ Highlight Caret Line, Color: (empty box).
- Externally Modified Files:** ☒ Always Prompt, ☐ Automatically Reload, ☐ Automatically Ignore.
- Indent:** ☐ Show Indent Guide, ☐ Insert Spaces Instead of Tabs, Indent Width: 4, Tab Width: 4.
- Page Guide:** ☐ Show Page Guide, Column: 80.
- Whitespace:** ☒ Invisible, ☐ Always Visible, ☐ Visible After Indent, ☐ Show End-of-Line.
- Margins:** ☒ Line Number, ☒ Bookmark, ☒ Fold.

• **Default Style: Font** 下拉列表控制源代码编辑窗口的字体，内含系统支持的各种定宽字体，**Size** 控制字体大小。如果 **Antialias** 多选框被勾选，字体显示平滑化。本区域设置的默认字体，通过 **View>Zoom** 菜单选项，可以针对文件对字体进行设置。

• **File Mode: Encoding** 选择保存源文件的编码方式，**Line Endings** 设定行结束符。Understand 支持多种编码方式。“system”使用与操作系统相同的编码方式，当某种应用程序打开 Understand 的文件遇到问题时，可以更改该选项进行尝试。默认情况下，这两个设置只对新创建的文件生效，包括文本文件和 CVS 文件，既有文件的编码方式不会被修改。如果勾选 **Convert existing line endings** 多选框，所有文件格式将按照此处设置进行保存。

- **Windows** 使用回车 (\r) /换行 (\n) 组合作为行结束符，也被称作 CR/LF。
- **Unit** 使用换行作为行结束符。
- **Classic Macintosh** 使用回车作为行结束符。

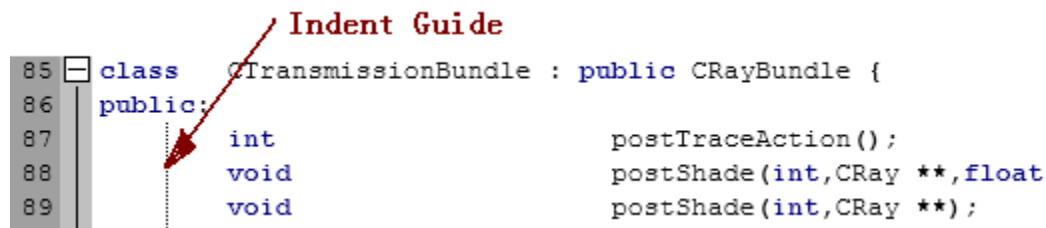
勾选 **Convert tab to spaces**，保存文件时 tab 被替换成 Width 中设置的数目的若干空格。勾选 **Add newline at end of file if absent** (默认勾选)，保存文件时在文件尾补充换行符。勾选 **Remove trailing spaces**，行尾的 tab 和空格会被自动删除。

• **Caret Line:** 勾选 **Highlight Caret Line** 使光标所在行高亮显示，默认颜色为亮灰，可以通过下面的 **Color** 进行设置。

• **Externally Modified Files:** Understand 2.5 能够检测到打开文件被外部程序修改的事件，勾选 **Always Prompt**，事件发生时 Understand 2.5 弹出提示框以获取下一步动作；**Automatically**

**Reload** 直接重新装载文件；**Automatically Ignore** 由于其危险性不推荐使用。

- **Indent:** 勾选 **Show Indent Guide** 可以通过点虚线显示各行的缩进状况。

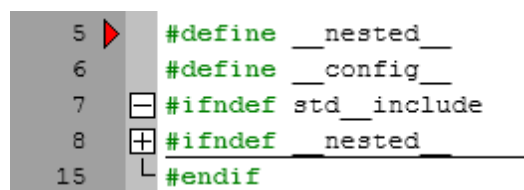


默认情况，**Insert Spaces Instead of Tabs** 选项取消，选中情况下输入<tab>会向源文件插入多个空格。Indent Width 设置一级缩进对应的空格个数，Tab Width 设置一个 Tab 对应的空格个数。例如，将 Tab Width 设置成 4，每次<tab>在光标右侧插入 4 个空格，同时将 Indent Width 设置成 6，每级缩进则有一个<tab>和 2 个空格组成。更多的缩进选项参见 **Editor>Advanced**。

- **Show Page Guide:** 勾选 **Page Guide** 可以以类似缩进指引的线条指示设置的行宽度（代码的右边界），Column 用于设置行宽度。

- **Whitespace:** 该选项用于控制空格或者 tab 的显示，点表示空格，箭头表示 tab，可选项包括 **Invisible**（默认），**Always Visible** 和 **Visible after Indent**。勾选 **Show End-of-Line** 可以显示换行符。

- **Margins:** **Line Number**（默认选中）用于控制行号显示；**Bookmark**（默认选中）用于控制标记显示（行号旁边的红色箭头）；**Fold**（默认选中）用于控制代码按照实体折叠/展开的能力。



## Editor>Advanced

选择 **Tools>Options**，打开 **Option** 对话框，展开 **Editor** 分类，选择 **Advanced**，可以对代码编辑器的行为进行进一步的设置。

以下选项控制编辑器窗口打印显示。

- **Font Size:** 选择打印输出的字体大小，针对单个文件的放大缩小参见 6-16。

- **Color Mode:** 设置打印输出的颜色模式，只有使用彩色打印机并且对打印驱动进行合适配置时才能进行彩色打印，选项如下：

- “Normal” 按照屏幕显示打印。
- “Invert Light” 反色打印，当背景色设成深色，文本设成亮色时选用。
- “Black on White” 以白色背景黑色字体打印。
- “Color on White” 以白色背景彩色字体打印。

Print

Font Size: 10

Color Mode: Normal

Wrap Mode: Wrap Word

Auto-complete

☐ Enable Auto-complete

☒ Automatically suggest matches

☒ Ignore case

Auto-indent

☒ Enable auto-indent

☒ Indent after newline

Tab indents: Never

Trigger characters: :#{}

Indent bare braces: 8

Vertical Caret Policy

☒ Even

☒ Jumps

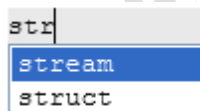
☒ Strict

☐ Slop

Slop Value 4

• **Wrap Mode:** 设置打印输出的包装模式，默认情况下编辑器始终将完整的单词放到一行，选择行截断或者字符级别包装，在行末的单词可能由于其长度会被分散放到不同行。行截断仅仅对打印显示生效，源代码文件的单词并不会受其影响。屏幕显示的包装模式参见行包装。

**Auto-complete** 区域对编辑器输入的关键词和实体名称的自动补全选项进行设置。输入时，补全单词在输入下方显示，通过上下箭头选择期望输入，点击 **Enter** 确认选择。



- **Enable Auto-completion:** 默认该选项取消，勾选可使能自动补全。
  - **Automatically suggest matches:** 勾选该选项，推选单词自动显示在输入下方。取消该选项，输入时通过 **Esc** 也能够使用自动补全功能。
  - **Ignore case:** 勾选该选项，推选单词忽略大小写。
- Auto-Indent** 区域由于控制 **tab** 自动插入。勾选 **Enable auto-indent**，代码编辑器输入时自动缩进使能，该特性目前已经支持 **C/C++**，**C#**，**Java**，**Javascript** 以及 **Perl** 代码编辑。
- **Indent after newline:** 勾选该选项，新建一行时，**tab** 被自动插入以确保首字符位置与前行保持一致。否则，新行首字符始终从第一列开始。
  - **Tab indents:**。如果选择 **Never**（默认选中），**<Tab>**键输入始终插入 **tab** 或者空格字符。如



果选择 **Always**，<Tab>键输入会自动选用合适的缩进层次。如果选择 **Leading Whitespace**，<Tab>键输入在行首空格使用合适数量的 **tab** 进行缩进，其他地方按照输入插入 **tab** 或者空格字符。

- **Trigger characters:** 输入特定字符时，Understand 基于代码分析自动选择正确的缩进层级。例如，“{” 增加一个缩进层次，而 “}” 减少一个缩进层次。使用 **Ctrl+Z** 可以撤销之前发生的自动缩进。默认的缩进触发字符包括 “#” “:” “{” “}”。

- **Indent bare braces:** 该值被设置成大于 0 的值时，如下面例子中 **Indent bare braces** 设成 2，**Indent width** 设成 4，自动缩进能够对代码进行格式化。

```
If (foo)
{
    do_foo();
}
```

**Vertical Caret Policy** 区域用于设置源代码编辑器的滚动条控制文本光标或者高亮显示当前位置的上下移动。通过这个区域可以控制当前位置附近的上下文内容显示最优化。一般情况下没有必要对这些设置进行修改，如果希望获取本区域选项的设置效果，可以访问 [http://www.scintilla.org/ScintillaDoc.html#SCI\\_SETYCARETPOLICY](http://www.scintilla.org/ScintillaDoc.html#SCI_SETYCARETPOLICY)。

- **Even:** 勾选此选项，设置上下滚动的动作保持一致。
- **Jumps:** 勾选此选项，设置代码根据需要滚动多行以显示当前行的上下文。
- **Strict:** 勾选此选项，禁止光标进入 **Stop** 设置的区域。如果 **Stop** 为非选中状态，上下滚动过程中始终保持当前行位于屏幕中央位置。
- **Stop:** 通过此选项以行号为标识设置禁止光标进入的编辑器的上下区域。
- **Stop Value:** 此选项用于设置禁止光标进入的编辑器的上下区域的行号。

## Editor>Styles

选择 **Tools>Options**，打开 **Option** 对话框，展开 **Editor** 分类，选择 **Styles**，可以对代码编辑器的颜色显示进行定制。

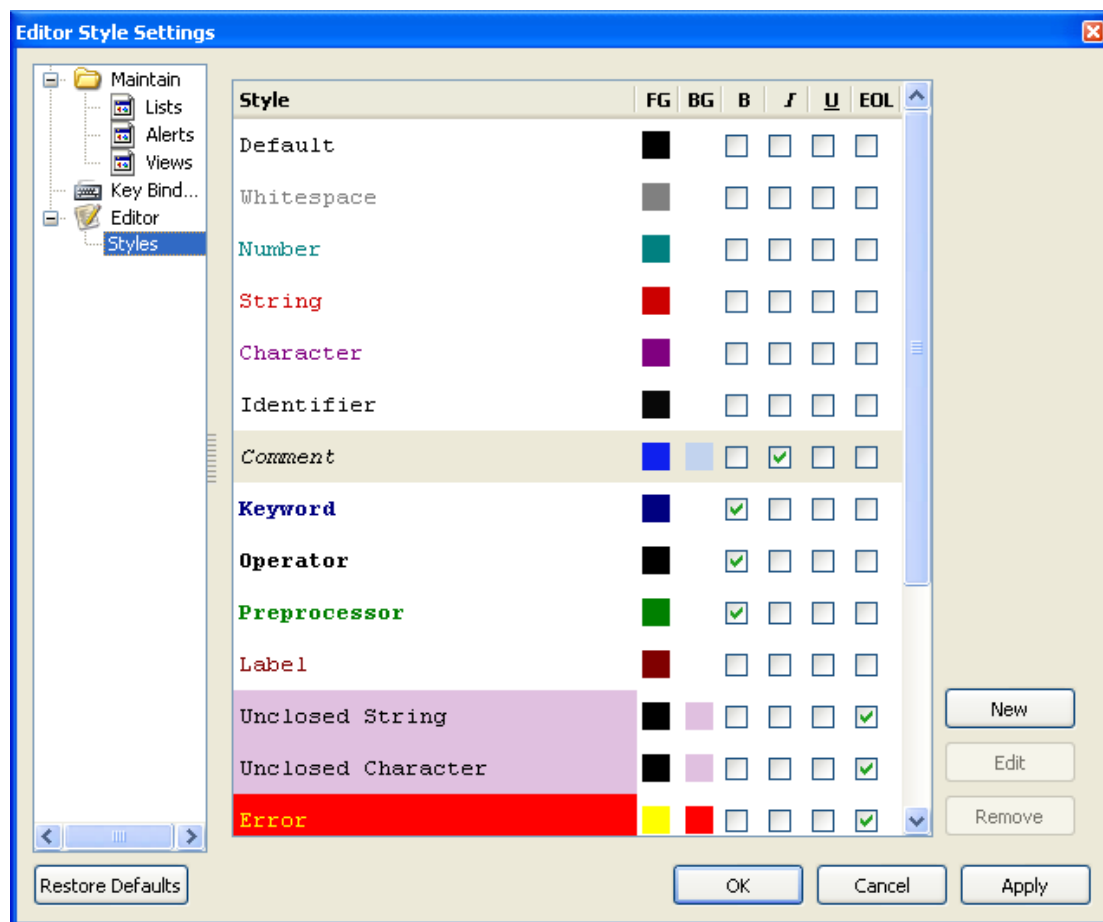
点击列表某项旁边的色块区域，通过颜色选择对话框即可修改对应项的颜色。

每项的字体色和背景色都可以被修改，字体还可以设置成加粗（**B**），斜体（**I**）或者下划线（**U**）以及高亮显示（**EOL**）。

默认情况下，源代码使用以下颜色设置：

- **Dark blue text:** 语言关键词。
- **Red text:** 字符或者字符串。
- **Italic blue text:** 注释。
- **Green text:** 预处理语句。
- **Black text:** 行号以及代码其他内容。
- **White background:** 大部分代码区域。
- **Pink background:** 非活动代码区域。
- **Gray background:** 行号区域。
- **Yellow background:** 查找结果的高亮显示。



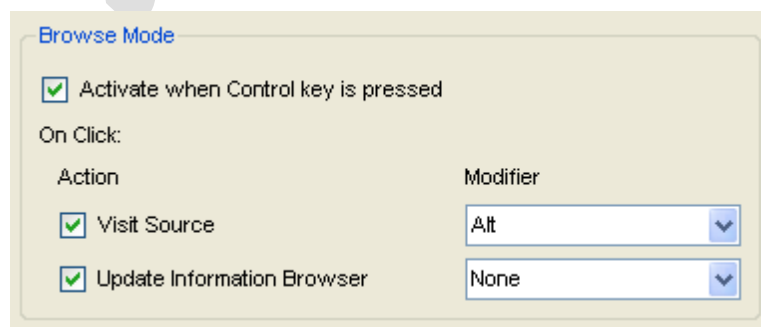


此外，根据工程使用的语言类型不同，还有额外的项目颜色可以被设置。如 Delphi，可以对模块，过程和类型名称的颜色进行设置；FORTRAN 可以对块，模块，子程序和类型名称进行颜色设置；Ada 可以对包名称，子程序名称和类型名称进行颜色设置。

点击 **New**，在弹出的用户风格对话框，输入名称及相关的语言类型，设置需高亮显示的关键词，可以新建一个自定义的风格分类。关键词以空格，换行或者 **tab** 分隔。设置后点击 **Save** 进行保存后，即可按前面介绍设置该风格的颜色。

## Editor>Browse

选择 **Tools>Options**，打开 **Option** 对话框，展开 **Editor** 分类，选择 **Browse**，可以对代码编辑器的浏览模式进行控制。



• **Activate when Control key is pressed**: 勾选此选项（默认选中），当保持控制键激活指向一个实体时，代码窗口使用浏览模式。

• **Visit source:** 勾选此选项（默认选中），浏览模式下点击一个实体，编辑器跳转到该实体的声明语句。默认 **Alt** 键用于控制跳转，也可以设置成 **none** 或者 **Shift**。

• **Update Information Browser:** 勾选此选项（默认选中），浏览模式下点击一个实体，信息浏览器显示该实体的相关信息。默认不需要使用其他控制键，也可以修改成使用控制键。

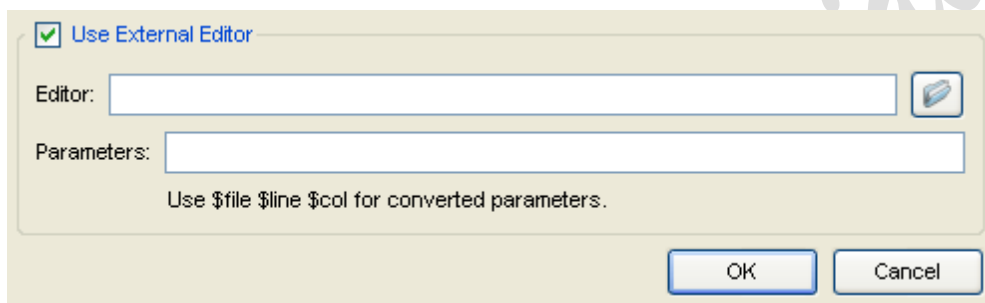
## Editor>External Editor

Understand 2.5 支持使用外部编辑器对代码进行浏览和编辑，对于习惯于使用单一浏览器的用户尤其方便。例如，用户可以选择使用 Micro visual C++或者 Emacs 作为 Understand 2.5 的编辑器。

对于外部编辑器，Understand 要求该编辑器支持使用命令行对文件进行打开和行列定位。

按照以下步骤设置外部编辑器：

- 1 选择 **Tools>Options**，展开 **Editor**，选择 **External Editor** 分类。
- 2 在外部编辑器设置对话框，勾选 **Use External Editor** 激活使用外部编辑器。



- 3 在 **Editor** 输入框，点击文件夹图标选择期望使用的编辑器可执行文件。

- 4 在 **Parameter** 输入框输入该编辑器使用的打开文件需要使用的命令行参数，使用 **\$file**，**\$line**，**\$col** 确保外部编辑器能够正确的打开文件。

例如，对于 UNIX 系统的 GVIM 编辑器，Editor 输入 “gvim”，Parameter 按照如下设置（GVIM 6.0 以及更新版本）：

```
--servername UND --remote +$line $file
```

对于 Windows 的 TexPad 编辑器，Editor 为“c:\Program Files\textpad4\textpad.exe”，Parameter 如下：

```
$file($line,$col)
```

外部编辑器还能够使用 Understand 2.5 的右键菜单（又称为上下文菜单），SciTools博客上介绍了设置方法。Emacs，vi和Visual Studio参见

<http://scitools.com/blog/2008/08/understand-context-menu-in-ema.html>，

SlickEdit参见

<http://scitools.com/blog/2008/05/using-understand-with-an-exte-2.html>。

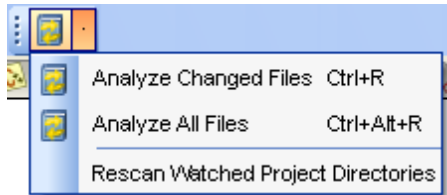
## 代码分析

对工程进行设置后，Understand 2.5 就可以对该工程进行分析，通过分析，所有源文件被检查，相关数据被保存到 Understand 2.5 的数据库。经过分析后，Understand 2.5 通过数据库对整个工程进行浏览。

工程配置被修改或者保存时，工程分析的提示框自动弹出。可以选择如下方式对工程进

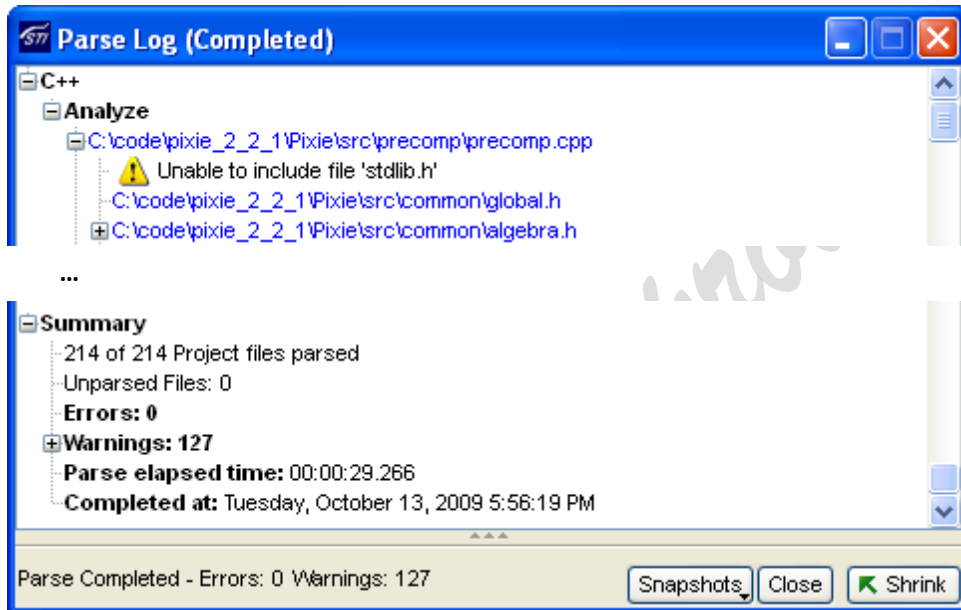
行分析。

**Project>Analyze Changed Files:** 此种分析方式对自上一次分析以来修改过的文件，以及依赖这些文件的其他文件进行分析，被称为“增量分析”。进行增量分析，也可以通过左击所图所示的工具栏图标（Ctrl+R）触发。



**Project>Analyze All Files:** 此种方式对整个工程的文件进行完整分析，其快捷方式为右击图标，选择 **Analyze All Files**（Ctrl+Alt+R）。

两种方式都通过状态栏显示分析状态，通过命令行报告窗口显示报告日志。



对于大型工程的分析，耗时会比较长。分析过程中点击 **Cancel**，会弹出窗口提示该操作会使工程处于非完整状态，用户可以在这些时候重新分析工程以解除这种状态。

分析日志显示了不可分析的文件，错误和告警。

如果工程分析产生了不希望的错误或告警，可能需要重新检查工程配置对话框以确定相关设置是否正确。多个类似的告警大多出于一个原因，如没有包含文件路径缺失。

分析结束后，双击分析日志窗口的信息可以对产生错误或告警的源代码进行检查。点击窗格分割箭头，可以在日志窗口中显示双击的错误的相关代码。

右击分析日志的白色背景区域，选择 **Save As**，选择期望的保存路径和文件名，可以将分析日志保存到指定的文本文件。也可以使用 **Copy to Clipboard**，将日志内容保存到剪贴板，拷贝到其他应用程序。

使用 **View>Last Parse Log** 可以直接打开上一次的解析日志。

#### 小技巧:

通过命令程序“und”可以以批处理模式对工程进行分析，参见使用 und 命令行。

控制分析过程参见 [Analyze 分类](#)。

安排分析动作自动执行，参见[计划活动](#)。

---

## Understand 1.4 工程转换

略

Shiningstone provided

---

## 代码库浏览

本章详细介绍 Understand 2.5 的基本窗口和相关选项，以及过滤区域和信息浏览器的相关操作。

用于查找和实体定位的文件查找和实体定位器的操作使用的相关详细信息参见第 5 章的代码查找。

源代码编辑器的使用详细信息参见第 6 章的代码编辑。

本章包含以下内容：

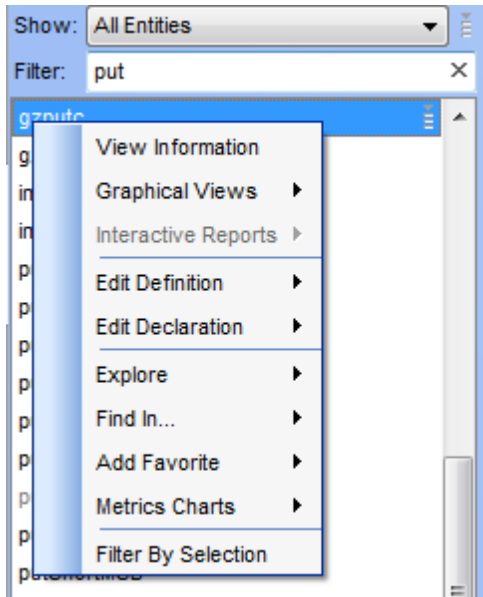
- 右键菜单
- 窗口介绍
- 实体过滤器
- 信息浏览器
- 工程浏览器
- 层级浏览
- 依赖浏览器
- 收藏夹

Shiningstone provided

## “右键菜单”

本章节标题与其他所有章节略有不同（使用了双引号标注），为了使 Understand 2.5 用起来更快速便捷，鼠标右键下隐藏了许多强大的快捷功能。

Understand 2.5 的“潜规则”就是任何地方都可以使用右键菜单获取相关信息或者执行进一步操作。不同类型的窗口中右键菜单的内容各有千秋，使用 **ctrl+鼠标右键** 还可以创建新的窗口执行相关动作。所以，本文其他地方不再对右键菜单的操作进行提示，尽情地右击吧！



## 窗口介绍

Understand 2.5 图形用户界面提供了大量定位和检查实体的工具。本章节简单介绍这些工具，并对实体过滤器，信息浏览器和收藏夹进行了详细描述。

用于查找和浏览实体的工具包括：

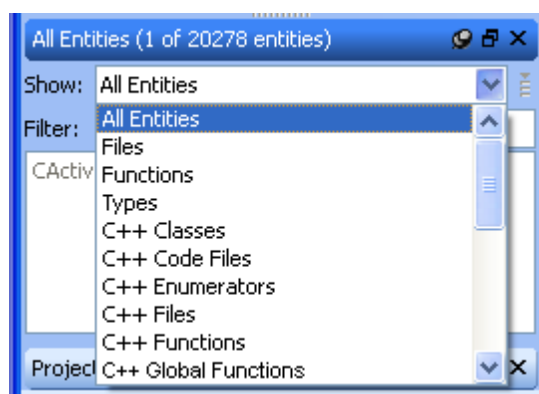
- 实体过滤器：按照字母顺序列出所选类型的实体，参见。
- 信息浏览器：显示实体的特征和联系，参见。
- 工程浏览器：显示层次化的文件列表，参见。
- 浏览视图：显示关联层次，参见。
- 依赖浏览器：显示依赖关系，参见。
- 收藏夹：提供快速选中常用实体，参见。
- 实体定位器：以复杂方式过滤实体，参见。
- 文件查找：查找多个文件，参见。
- 代码编辑器：源代码编辑，参见。
- 上下文信息边栏：显示当前代码文件的上下文信息，参见。
- 审视列表：列出文件中的函数或者相似结构体，参见。
- 架构：为工程定义命名区域和命名视图，参见。
- 图形视图：显示实体的联系和结构，参见。
- 报告：生成实体报告，参见。

- 度量：生成实体度量数据，参见。

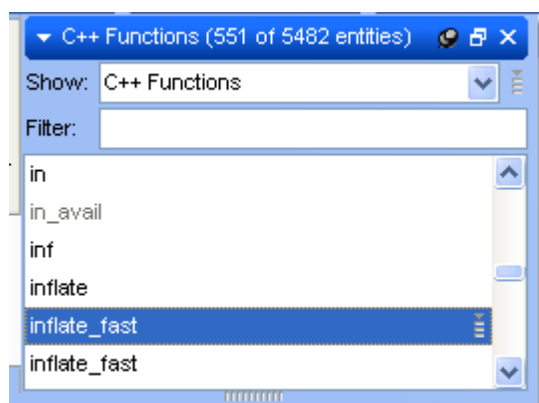
## 实体过滤器


实体过滤器提供选中实体类型的列表，通过字符串在列表中进行快速查找。

Show 下拉列表框中的内容根据当前工程选择的语言类型以及相关的实体和关联关系有所不同，如果工程选择了多种语言，该列表还会包含语言相关的实体条目。



针对各种分类，用户可以快速定位实体对应的代码位置。




默认情况下，实体根据名称以字母升序(A到Z)排列，通过点击  选择 **Sort Descending** 可以修改成按照降序排列。

实体过滤器仅支持一个实例窗口，被关闭后，通过 **View>Entity Filter** 可以重新打开。

## 过滤区使用

在 Filter 区，通过输入一个字符串查找匹配的实体，匹配是指该实体名包含所输入的字符串（不限定位置）。例如，输入“y”可以显示出所有名称中包含 y 的实体。


默认情况下，过滤器不区分大小写。点击  选择 **Filter Case Sensitivity>Case Sensitive** 可以控制过滤器对大小写进行区分。

点击列表框，输入一个字符，过滤器会自动跳转到以该字符开头的第一个实体。

Filter 区的工作方式可以通过  中的 **Filter Pattern Syntax** 进行修改，以下方式可供选择：

- **Fixed String:** 默认方式。
- **Wildcard:** 支持\*（任意字符）和?（任意单个字符），示例参见。
- **Regular Expression:** 支持 UNIX 风格正则表达式，示例参见。

当结束一个过滤查找，点击  选择 **Clear Filter** 可以重新显示所有实体。

如果期望对 **Show** 区域的实体类型进行修改时自动清除过滤器输入，可以点击  选择 **Clear Filter Text on Filter Type Changes**。

## 显示定制

实体过滤器的显示可以根据需要进行定制：

默认情况下，实体过滤器列表显示实体全名，并且按照字母顺序对实体进行排列。实体全名有可能包含如类名前缀或者其他具有语言特征的前缀，通过下拉菜单选择 **Entity Name as>Short Name**，可以仅显示实体的短名称。

默认情况下，实体过滤器的文件列表仅显示文件的名称，而不包括其路径。通过下拉菜单选择 **File Name as>Relative Name** 或者 **File Name as>Long Name** 可以显示文件路径。

## 根过滤器

Understand 2.5 提供了若干名称包含“根”的过滤类型，如根调用，根被调，根包含等，这些类型限定相关树的末端实体，对于阅读新的代码，凸显出一个关系树的顶端(或者末端)通常尤其有用。

- **Root Calls:** 仅显示调用而不被调用的实体，如上层代码（**mains**），硬件调用的代码（中断处理函数）以及哑（不被使用）代码。
- **Root CallBys:** 仅显示只被调用而不调用其他实体的实体，即最底层的函数。
- **Root IncludeBys:** 仅显示被其他文件包含的文件，即基础头文件。
- **Root Classes:** 仅显示不依赖于其他类的类，包括底层类和库类。
- **Root Decls:** 仅显示最高层的声明过程（**Ada**）。
- **Root Withs:** 仅显示被其他单元使用，而不使用其他单元的程序单元（包括包，任务，子程序等）（**Ada**）。

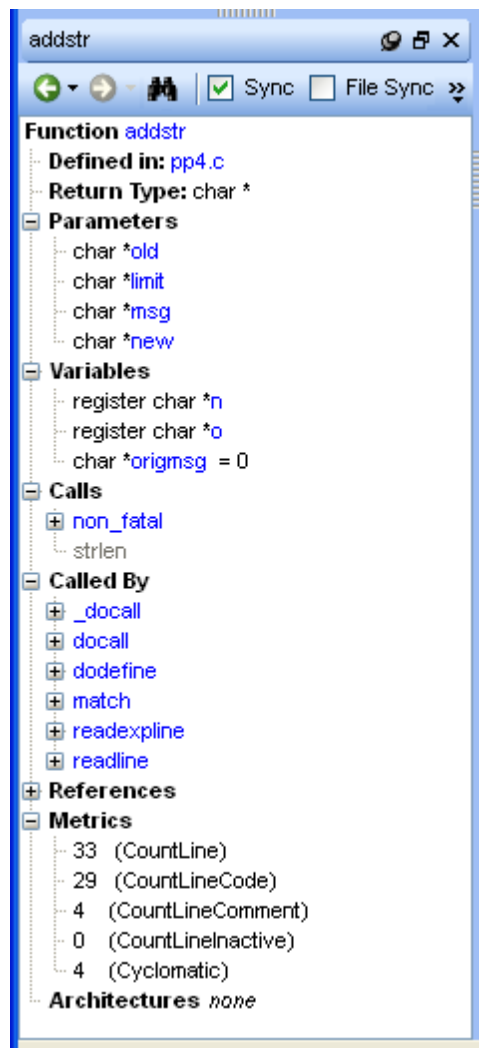
## 信息浏览器

点击实体过滤器或者大多数其他窗口中的某个条目，信息浏览器会显示相关实体的相关信息，这些信息以树的形式呈现，分支可以单独控制，也可以统一操作。

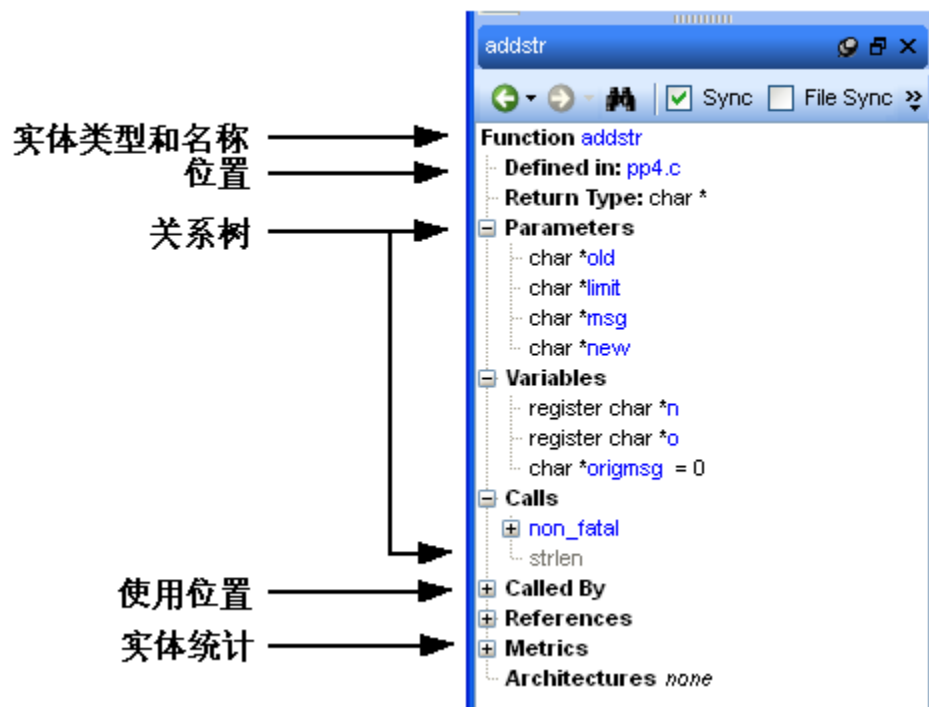
点击实体过滤器中的一个条目或者工程浏览器，即可打开关闭的信息浏览器。右击任意一个条目，选择 **View>Information**，或者选择菜单栏上的 **View>Information Browser** 也可以打开该窗口。

通过信息浏览器可以了解 Understand 2.5 对于某一实体的所有信息，这些信息以树的形式呈现，分支可以单独控制，也可以批量操作，树的每个叶子显示了该实体某方面的信息。这些信息可以保存到文本文件，也可以通过标准的 **Windows** 或者 **X11** 拷贝动作进行拷贝和粘贴。



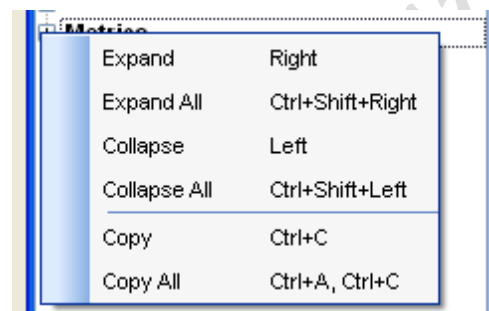


对代码进行深度阅读时，用户可以随意切换关注的实体，所有浏览过的实体都会被记录以便回退浏览。




## 深度关系浏览

通常用户都会对信息树进行深度了解，展开分支使用“+”，合并分支使用“-”。右键菜单中的 **Expand All** 展开所有分支，**Collapse All** 合并所有分支。右键菜单中其他选项的详细信息参见保存和打印信息浏览器内容。



## 信息显示内容控制

点击信息浏览器中的粗体标题，如 **Calls**，**Called By** 或者 **References** 旁边的 （或者右击这些标题），可以对实体显示方式进行设置。其中包括：

- **Full Name**: 显示实体全名。
- **Parameter**: 显示参数。
- **Reference**: 选择“Full”可以显示引用所在的文件和行。
- **Return Type**: 显示返回值。
- **Sort**: 排序方法。

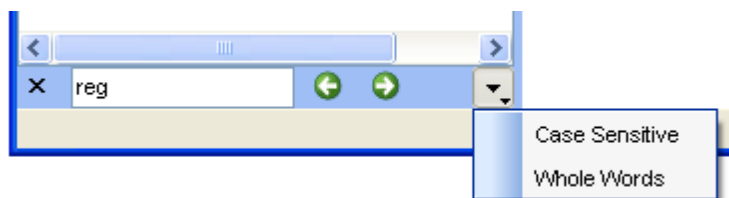
- **Type:** 显示数据类型。
- **File Name:** 控制以短名称，长名称或者相对名称方式显示引用。
- **Group By:** 控制 C++ 类成员以访问权限（公有/私有）或者成员类型（函数/对象）分组显示。

## 信息浏览器查找

点击信息浏览器上方的望远镜图标，或者点击浏览器后按 **Ctrl+F**，信息浏览器下方会显示一个搜寻栏。

在搜寻栏中输入字符，点击前进或后退，可以在信息浏览器显示内容中逐个查找，范围包括所有内容，如节点名称和被关闭分支中当前不可见的条目。如果没有找到对应内容，搜寻栏会变成红色。

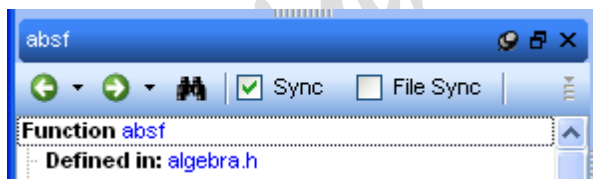
使用下拉箭头选择 **Case Sensitive** 和 **Whole Words**，可以控制查找进行大小写匹配和全字匹配。



## 信息浏览器同步

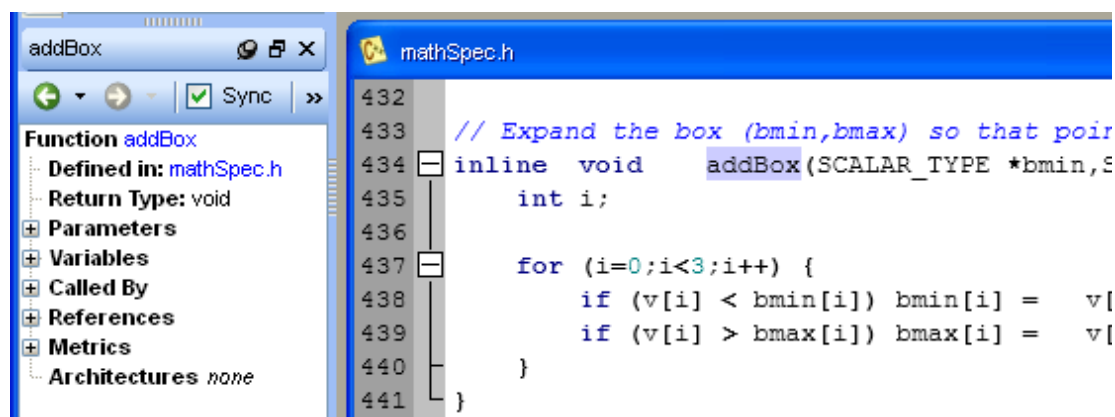
取消 **Sync** 勾选可以打开多个信息浏览器窗口，**Sync** 勾选的情况下，选中一个实体或者选择 **View Information** 则会更新当前信息浏览器。

选中 **File Sync** 可以保证信息浏览器内容始终与激活的代码编辑器保持同步。



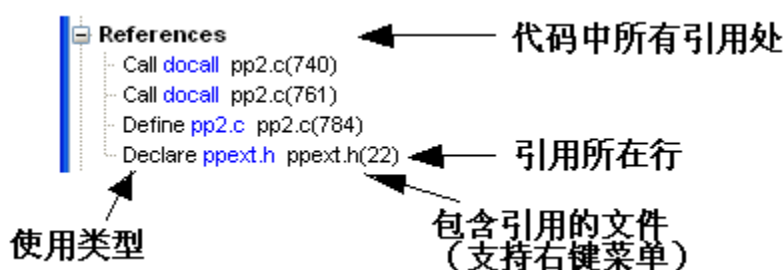
## 源代码查看

通常情况下，双击信息类窗口（如信息浏览器或者实体过滤器）中的一个实体，文档区域就会自动载入对应实体的声明代码。使用右键菜单也可以快速浏览所见实体的代码——在合适的区域，实体的右键菜单显示会包括 **Edit Source** 菜单项，根据具体情况，**Edit Definition** 和 **Edit Declaration** 会分别显示，甚至还包括语言特征的定位方式。

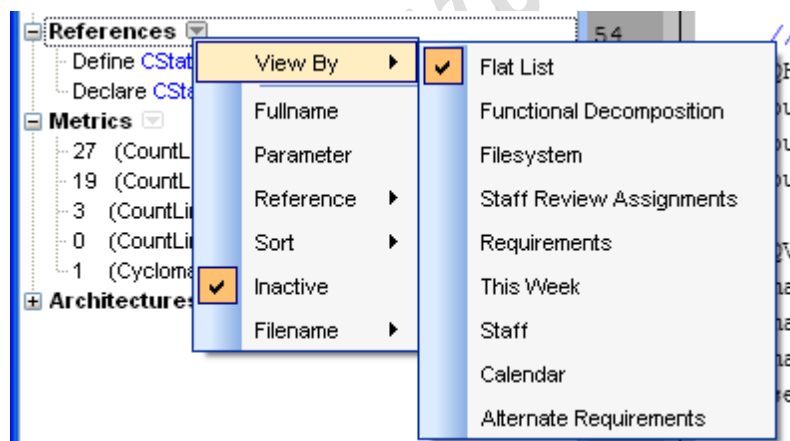


## 引用查看

信息浏览器的“References”区域显示了所有引用选中实体的相关信息：



左击引用可以跳转到对应的代码区域。右击节点使用右键菜单或者点击节点旁边的下拉箭头可以对节点所包含信息的显示方式，默认为扁平化列表。

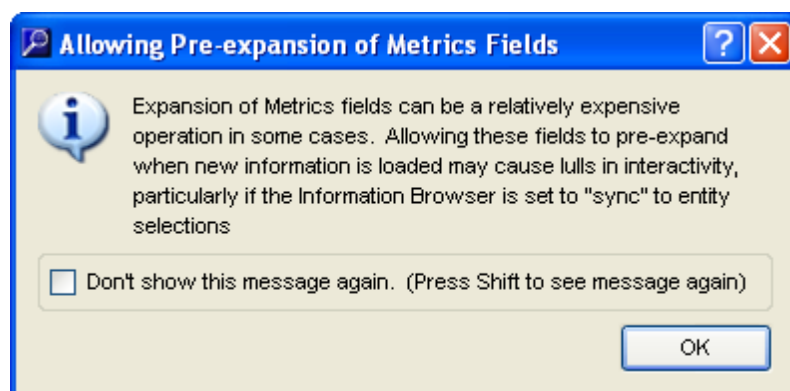


## 度量查看

信息浏览器的最后一个节点是 **Metrics**，显示的是选中实体的度量信息。

因为在大型工程中实体的度量信息更新通常需要花费一段比较长的时间，信息浏览器中的实体焦点发生变化时，该节点默认情况下会被自动关闭。如果工作在一个不大的工程，实体焦点变化时节点的度量信息更新花费时间不长，右击 **Metrics** 节点选择 **Allow Pre-expansion**，可以取消自动关闭，此时实体变化时 **Metrics** 节点始终保持展开状态。启用这一选项时，

Understand 2.5 会弹出如下提示窗口。



度量的详细介绍参见度量报告。

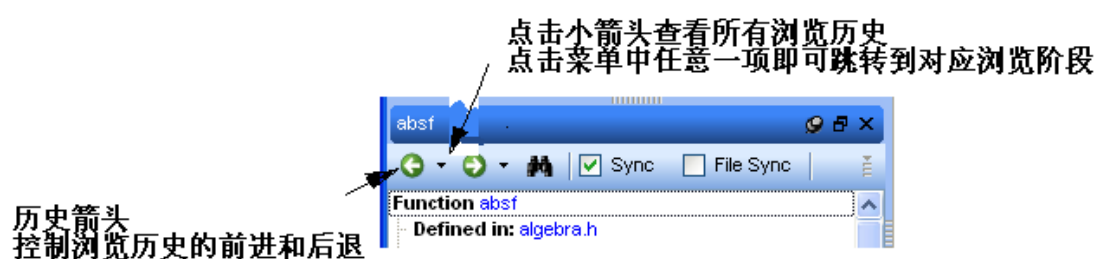
## 保存和打印信息浏览器内容

信息浏览器中显示的所有文本内容都可以以原文本的格式拷贝到剪切板，粘贴到其他应用程序。以文本格式保存或打印时，树的分支结构以缩进的方式体现。

右键菜单提供了 Copy 和 Copy All 选项。

## 实体浏览历史

使用 Understand 2.5 浏览代码过程中，可以快速跳转到许多关联区域。经常地，用户需要回退浏览，达到一个节点时，重新开始另一个路径的浏览，为此 Understand 保存了所有之前的浏览历史，并在窗口左下角显示出来。



使用左箭头和右箭头控制浏览历史的前进和后退，向下箭头显示所有浏览历史。

## 工程浏览器

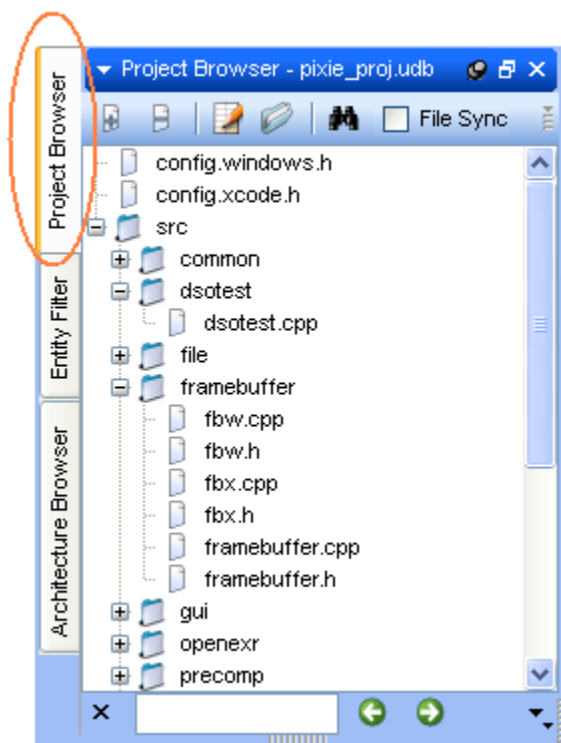
选择菜单栏上 View>Project Browser 打开工程浏览器。

默认情况下，工程浏览器与实体过滤器以及层级浏览器位于同一区域，通过左侧标签页在不同的浏览器工具之间切换。

工程浏览器通过树形结构按照目录层次显示工程中所有的文件，所有节点可以根据需要展开或关闭。

工程浏览器可以通过 Ctrl+F 在其底部开启一个查找输入区域。

File Sync 多选框控制工程浏览器显示与代码编辑器中的文件保持同步。



工程浏览器的操作选项可以通过工具栏上的按钮或者右键菜单进行。

### 在信息浏览器中显示相关信息

使用文件查找

展开所有分支

关闭所有分支

拷贝所有内容

拷贝选中内容

向工程添加一个文件

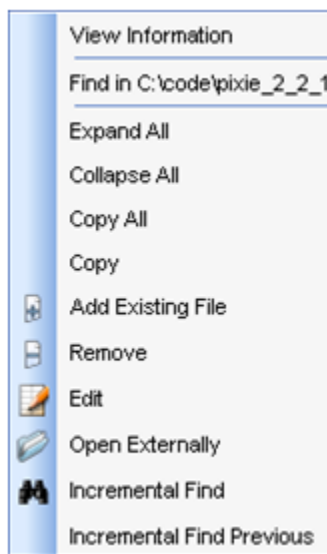
从工程删除选中内容

为选中内容打开编辑器

使用外部程序打开选中内容

在工程浏览器内部查找

工程浏览器内部反向查找



对于一个文件，右键菜单包含额外的选项，如打开图形视图，针对文件进行分析，查找使用该文件的工程文件，还有将该文件加到收藏夹。

通过 Add Exist File 向工程添加一个文件，通过 Remove 将一个或者多个选中的文件和文件夹从工程中删除，删除文件时会弹出工程变更确认对话框，点击 Yes 即可。

Open Externally 使用操作系统相关的应用程序对选中文件或者文件夹进行操作，如在 Windows 系统中选中一个目录点击该选项，Understand 会使用 Windows 浏览器打开该目录，而对于文件，Understand 会使用与该文件扩展名匹配的程序打开文件。

Incremental Find 会在信息浏览器底部开启一个查找框，输入文字，点击箭头可以在工程浏览器中的查找与输入相符的位置，查找范围包括工程浏览器中的所有内容，包括未被展开的文件夹中的文件。如果没有找到对应内容，查找框会变成红色。查找选项参见。

## 层级查看

Understand 2.5 的浏览视图针对工程中的关联层次关系提供了方便的浏览方法。



信息浏览器，实体过滤器和工程浏览器中的右键菜单提供了对某些实体类型的查看选项，类似于 **Explore>Explore Called By/Calls** 和 **Explore>Explore Includes**。

点击上图列表框中的某一项，可以通过左右两边的列表框查看该项的关联项，随着关注点的左移或右移，列表框大小发生变化以显示关联关系中更多信息。调用和包含的关系从左往右，被调用和被包含的关系从右往左。

双击其中某项，代码编辑器自动打开，并跳转到对应的实体定义区域。

References 区域显示了选中项所在文件的行号，双击可以跳转到对应代码。

勾选 **Generate Syncs** 多选框，信息浏览器内容会即时显示浏览窗口中的选中实体的相关信息，使用 **Shift** 键可以临时激活该项功能。

勾选 **Jump to First References**，代码编辑器自动跳转到浏览窗口中的选中实体的第一个引用处，使用 **Ctrl** 键可以临时激活该项功能。

## 依赖浏览器

通过依赖浏览器可以检查相互依赖的实体，这些实体包括层级节点，文件，类，包和接口。



右击 Understand 2.5 中如实体过滤器，信息浏览器或者图形视图区域中的层级节点，文件，类，包的名称，从上下文目录中选择 **View Dependencies**，即可打开依赖浏览器。

该浏览器的左侧面板显示选择的实体及其包含的所有条目，右侧面板则根据 **Dependency**



Kind 下拉列表的选中情况显示依赖该实体或者该实体依赖的条目。依赖是指某个实体包含，调用，设置，使用另外的实体。

左右面板显示的内容如果包含层次关系都可以被展开。例如，查看一个层级节点的依赖关系时，对其展开查看更底层的层级节点，进而到包含的文件，以及文件中定义的实体。显示条目旁边的字母表示了该条目所属的类型，如层级节点（a），文件（f），类（c）及实体（e）。




使用 Group Results By 下拉列表可以控制右侧面板内容的显示方式。

选中 Files 多选框，根据存在依赖关系的文件对右侧面板内容进行显示，文件位于最底层的层级节点下方。

选中 Classes 多选框，根据存在依赖关系的类对右侧面板内容进行显示，如果 File 和 Class 都被选中，类位于文件的下方。

选中 Entities 多选框，根据存在依赖关系的实体对右侧面板内容进行显示，如果 Class 和 Entities 都被选中，实体位于类的下方。

点击  图标可以对依赖浏览器中当前显示的依赖关系创建图形视图。与依赖浏览器相比，图形视图可以方便地显示依赖关系的多个层次，详细信息参见查看层级依赖图形。

点击  图标可以对依赖浏览器左侧面板的顶层条目创建一个 CSV 格式的报告。

依赖浏览器的右侧面板中的条目可以通过鼠标选中，右击字母图标选择 Copy 可以将选中文字拷贝至剪贴板供其他应用程序使用。

Reuse 多选框默认选中，不勾选的情况下，每个新实体或节点的 View Dependencies 操作都会打开一个依赖浏览器窗口。

Sync 多选框默认不选中，勾选的情况下，依赖浏览器即时显示在实体过滤器，工程浏览器，层级浏览器之类窗口中选中的层级节点，文件，类或实体的依赖信息。

下拉图标  控制如下显示选项：

- **Architecture Name:** 默认情况下层级节点名称显示为相对名称，也可以选择显示长名称或者短名称。

- **Entity Name:** 默认情况实体名称显示为短名称，可以修改成显示长名称。

- **File Name:**。默认情况文件名称显示为短名称，可以修改成显示长名称或者相对名称。

关于设定哪种关系为依赖的详细信息参见 [第 3 章 依赖分类](#)。

## 收藏夹

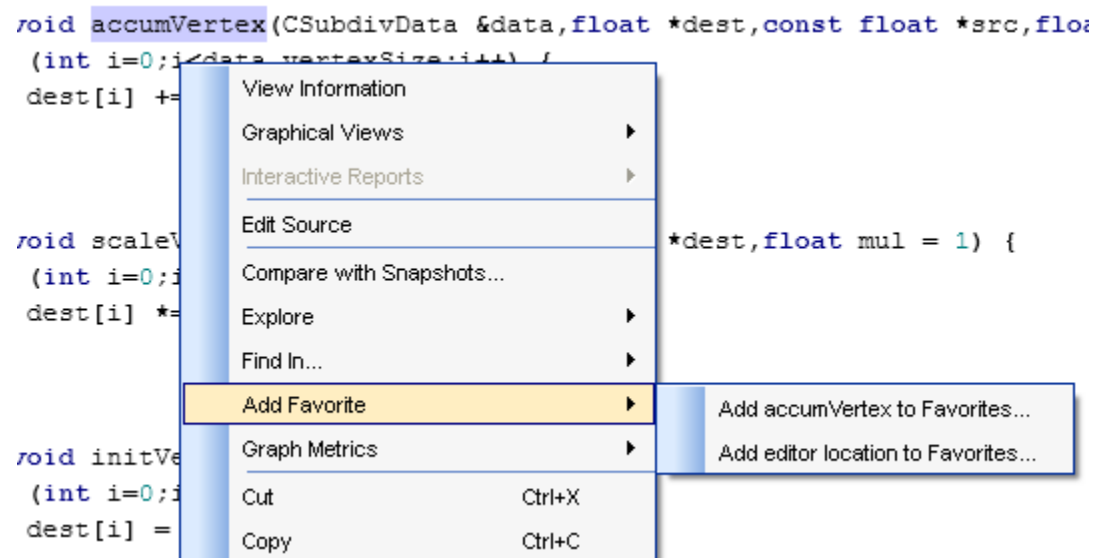
将实体标记为“Favorites”能够像 web 页面的收藏夹一样快速对其访问。



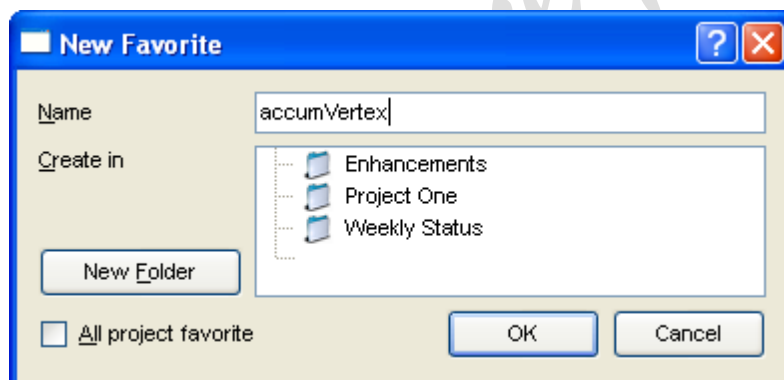
## 添加收藏夹

通过以下步骤将一个实体设置为收藏：

1 在任何显示实体的区域，如实体过滤器，信息浏览器，代码视图，图形视图，对一个实体单击右键。



2 在右键菜单选择 **Add Favorite**，弹出添加收藏对话框。如果是在代码区域操作，可以将对象选择为实体（不随行号发生变化）或者行号。



3 设置对应实体在收藏夹显示的名称。

4 在 **Create In** 区域，选择一个收藏文件夹，或者另外创建一个新的收藏文件夹。如果不指定收藏文件夹，操作实体会添加到该区域的“根目录”。

5 如果期望该收藏操作对所有工程生效，勾选 **All project favorite**。

6 单击 **OK** 完成。

通过菜单栏上 **Edit>Add Favorite** 也可以执行同样操作。

## 使用收藏夹

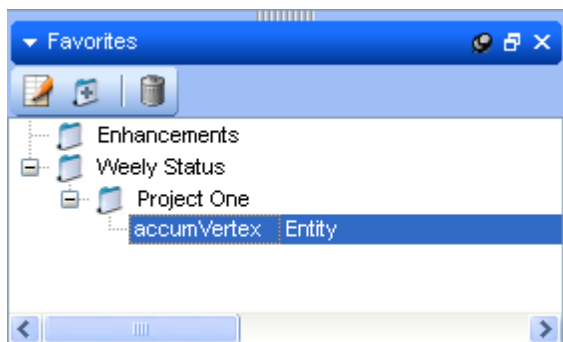
选择菜单栏上 **View>Favorites** 打开收藏夹，在收藏夹里可以执行以下操作：

- 双击一个条目跳转到代码中相应位置。
- 右击一个条目选择 **View Information**，在信息浏览器查看该实体对应信息（被只保存为代

---

码定位的条目除外)。

- 通过上方的 **Edit**, **Add**, **Remove** 图标对条目进行管理。



---

## 代码查找

本章介绍如何使用 Understand 2.5 提供的多文件查找和实体定位器来查找代码中的期望对象。

本章包含以下内容：

查找：概述

快速查找

多文件查找

实体定位器

查找窗口

Shiningstone provided

## 查找：概述

在大型工程的代码中查找一个对象，通常是困难冗长而且容易出错的，为此，Understand 2.5 提供了以下方案是用户从沉闷的等待中解脱出来：

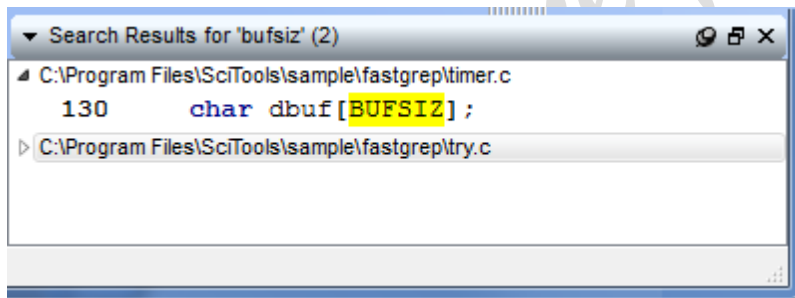
- 编辑器中的单个文件查找，参见查找代码。
  - 针对工程范围内文本内容的快速查找，参见快速查找。
  - 针对工程范围内文本内容的多文件查找，参见多文件查找。
  - 针对工程范围内实体的实体定位器，查找对象仅限于实体（不包括字符串，注释及没有遵循语法声明或者定义的内容），参见实体定位器。
  - 浏览器中的文本内容查找，参见[信息浏览器查找](#)和[工程浏览器](#)。
- 每种方案都各有优劣，它们互相结合为用户的查找和代码理解提供了强有力的支持。

## 快速查找

快速查找能够在上百万行代码规模的工程中进行快速查找。快速查找框位于 Understand 2.5 窗口的右上角，随着输入框内的输入变化，快速查找列表能够即时显示与当前输入匹配的内容。如果当前没有显示该输入框，可以选择菜单栏 **View>Toolbars>Search**，将其打开。

点击快速查找框即可触发快速查找，通过 **Ctrl+Alt+S**，或者选择菜单栏 **Search>Instant Search** 也可以将鼠标快速定位到快速输入框。

快速查找最常用的方法就是输入期望在代码中查找的内容后输入 **Enter** 即可在查找结果区域显示出包含查找内容的文件，双击文件可以查看到高亮显示的匹配位置。



此外，快速查找还提供了额外的查找选项，输入框语法基于开源文本搜索引擎库 Apache Lucene ([http://lucene.apache.org/java/2\\_3\\_2/queryparsersyntax.html](http://lucene.apache.org/java/2_3_2/queryparsersyntax.html))，以下列出了一些该语法选项：

- 大小写匹配，查找“test”能够匹配“Test”和“TEST”。
- 不使用通配符的情况下，查找进行全字匹配，查找“test”不匹配“testfile”
- 通配符包括“\*”（不定数目的字符和数字）和“？”（单个字符或数字），通配符不能位于查找内容首字符。

- 快速查找根据空格和标点符号（C/C++，Java，Ada 语法定义）将代码拆分成可查找的字符串，并据此索引（后台进行）。因此，在下面的代码中可查找字符串包括“foreach”，“1”和“10”。

```
foreach (i=1, i<10, i++)
```

- 快速查找不能对标点符号或者包含一般标点符号的字符串，如“i=1”，但是可以查找包含空格（使用引号标注）的字符串。
- 快速查找可以指定范围，如字符串，标识符和注释。默认情况下，查找范围包含上述三

---

种类型。例如，下面的查找仅限引号标注的字符串：

`string:test`

下面的查找仅限标识符如变量和函数名：

`identifier:test`

下面的查找仅限注释内容：

`comment:test`

- 快速查找支持查找内容的逻辑运算，默认情况下多个查找条目具有“或”的逻辑关系。因此，查找“for delta”等价于查找“for OR delta”。包含“for”或“delta”的文件被匹配。注意查找字符串用于匹配整个文件，而限于单独的声明区。

- 使用“for AND delta”查找同时包含“for”和“delta”的文件。
- 使用“+”可以在查找结果中继续查找指定内容，如下查找在所有包含“contain”的文件中查找包含“for”的文件。

`+delta for`

- 使用“-”或者 NOT 可以将查找结果中包含指定内容的文件移除，如下查找查找对象为包含“delta”和“delta1”而不包含“delta2”的文件。

`delta delta0 NOT delta2`

`delta delta1 -delta2`

- 快速查找支持包含圆括号的逻辑表达式，如

`(delta0 OR delta1) AND change`

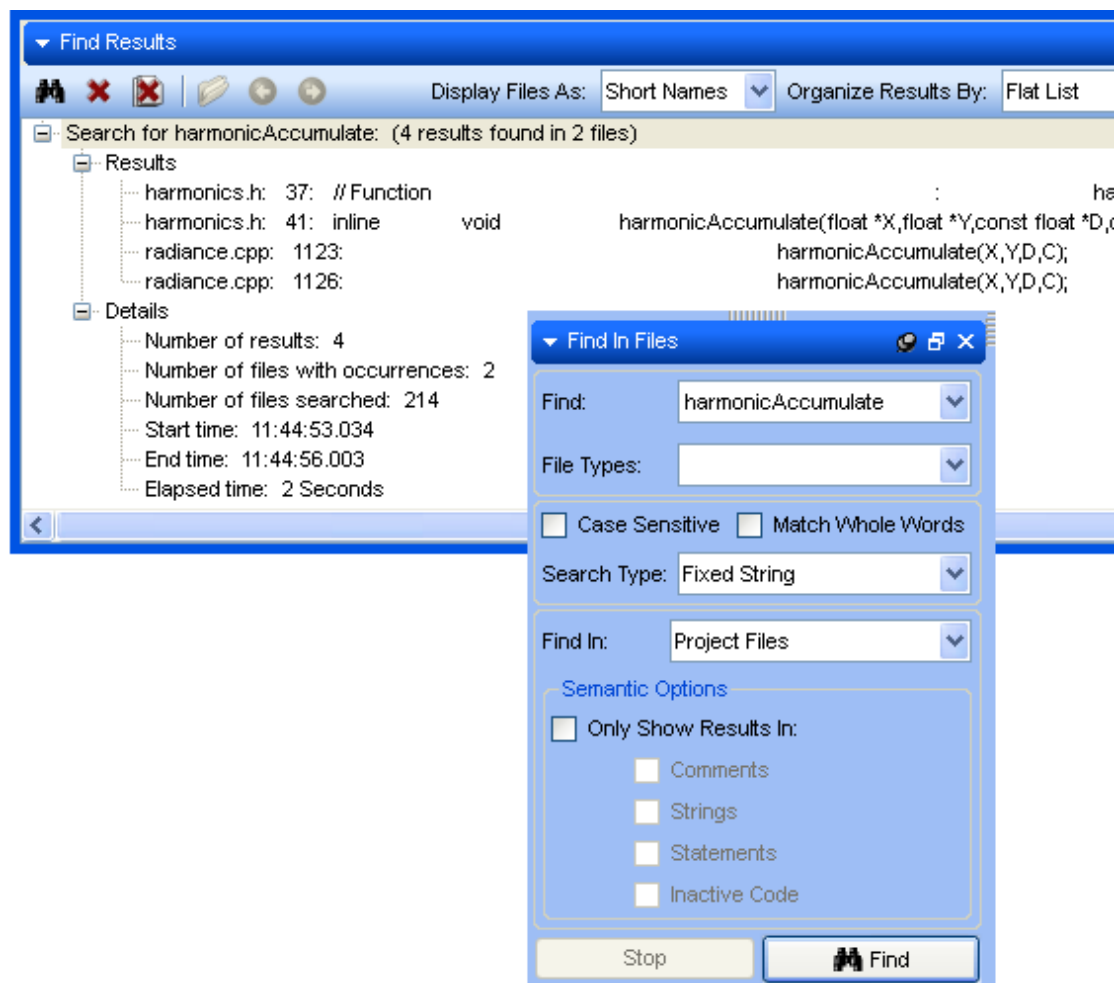
- 快速查找通过在字符串末加“~”支持模糊查找，如 boo~匹配 foo，too，book。

## 多文件查找

在菜单栏选择 **Search>Find in Files**，右键菜单中选择 **Find in...**或者使用 F5 均可打开多文件查找，在整个工程文件范围或者是这些文件的指定集合查找指定的字符串或者正则表达式。查找结果显示在查找结果窗口中，双击窗口中任意行即可跳转到代码中的对应位置。

本功能区定义了如下选项：

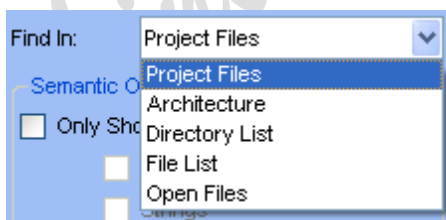
- Find**：输入期望查找的字符串，下拉列表中包含最近使用过的字符串。
- File Types**：通过文件扩展名设定查找范围，该输入域为空表示查找所有文件。如果 **Find In** 选项设为“**Open Files**”，本选项不可用
- Case Sensitive**：本选项控制大小写匹配，默认不匹配。
- Match Whole Words**：本选项控制全字匹配（“test”能够匹配“test”，不匹配“testing”），默认忽略单词边界。
- Search Type**：选择使用固定字符串，通配符或者正则表达式进行查找，详见。



• **Find in:** 选择查找范围-工程文件（所有或者只包含打开文件），指定层级节点下的文件，指定路径下的文件或者指定文件。

如果选择“Architecture”，“Directory List”或者“File List”，可以通过“+”对指定对象进行深度浏览。

右击代码或者其他区域的实体，选择右键菜单中的 Find In 选项，Find 和 Find In 自动获取选中字符串内容，快速对选中字符串进行查找。

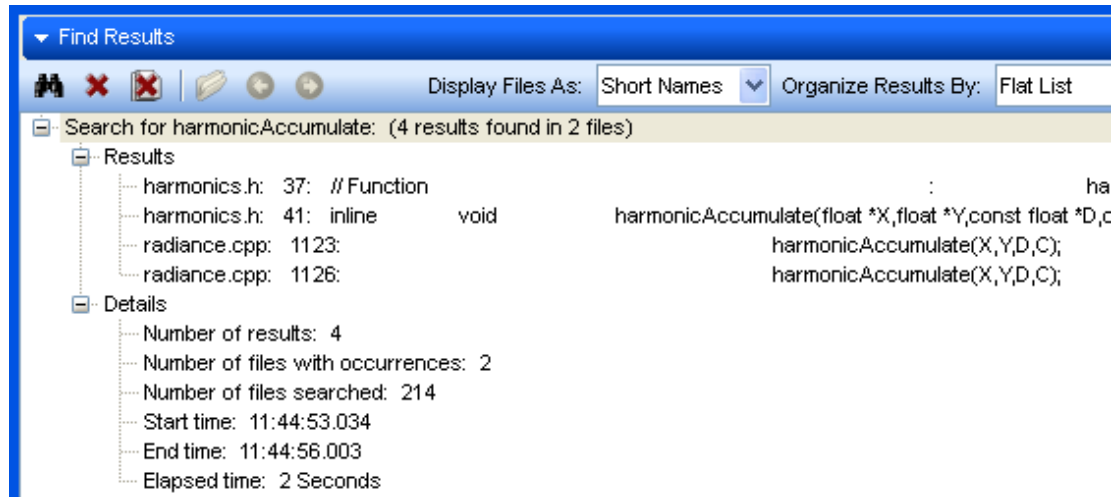


• 语义选项：如果选择“Project Files”查找，勾选 **Only Show Result In** 可以通过对复选框“Comments”，“Strings”，“Statements”，“Inactive Code”操作，选中一个或者多个，用以控制匹配项报告的内容。

确定查找条件后点击 Find，查找结果窗口即可显示所有匹配内容。如果觉得查找事件过长考虑修改查找条件，可以点击 Stop。

## 查找结果

查找结果窗口显示匹配内容以及相关的一些统计数据。**Result** 列表列出查找内容出现的行，**Detail** 区包含匹配个数，涉及的文件个数，所有查找过的文件以及此次查找所耗费的时间。



**Result** 列表可以显示多个查找的结果，右击窗口空白区域选择 **Expand All** 或者 **Collapse All** 可以展开或者关闭所有节点。

使用 **Orgnaize Results By** 下拉菜单可以对最近的查找结果显示进行组织，其选项包括扁平化列表（默认选项），基于文件的列表，使用默认层级结构或者自定义层级结构的层次化列表。

默认情况下，查找结果仅显示文件名称，通过 **Display Files As** 下拉列表选择“**Long Names**”可以显示出文件的完整路径。


双击查找结果中的匹配条目，即可使用代码编辑器打开相关文件，并且高亮显示匹配内容。


右键菜单中的 **Copy** 和 **Copy All** 可以将查找窗口中的内容拷贝粘贴到其他地方。


查找结果窗口的工具栏（或者右键菜单）支持以下操作：

 在当前选中的查找结果中查找（参见信息浏览器查找）。

 删除当前选中的查找结果。

 删除所有查找结果。

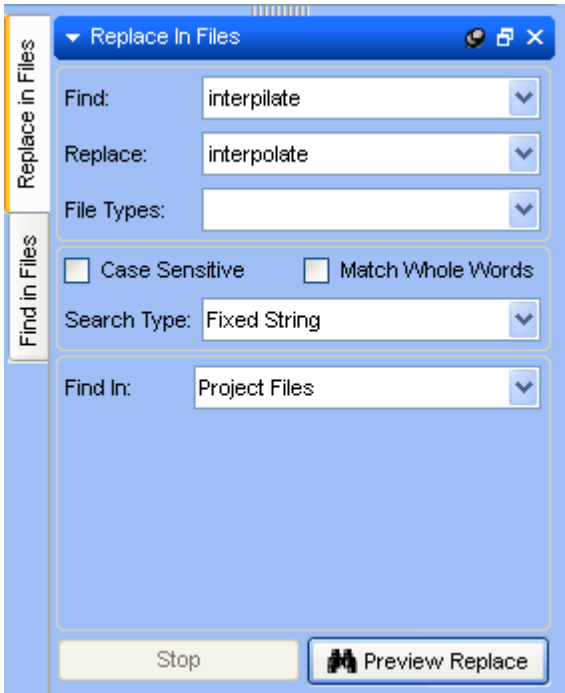
 在代码编辑器中打开选中结果。

 选中上一个或下一个匹配条目。

使用 **Search>Show Find Results** 可以重新打开查找结果窗口，除非对查找结果进行过删除，否则上一次查找的结果会在查找结果窗口中显示。

## 多文件替换

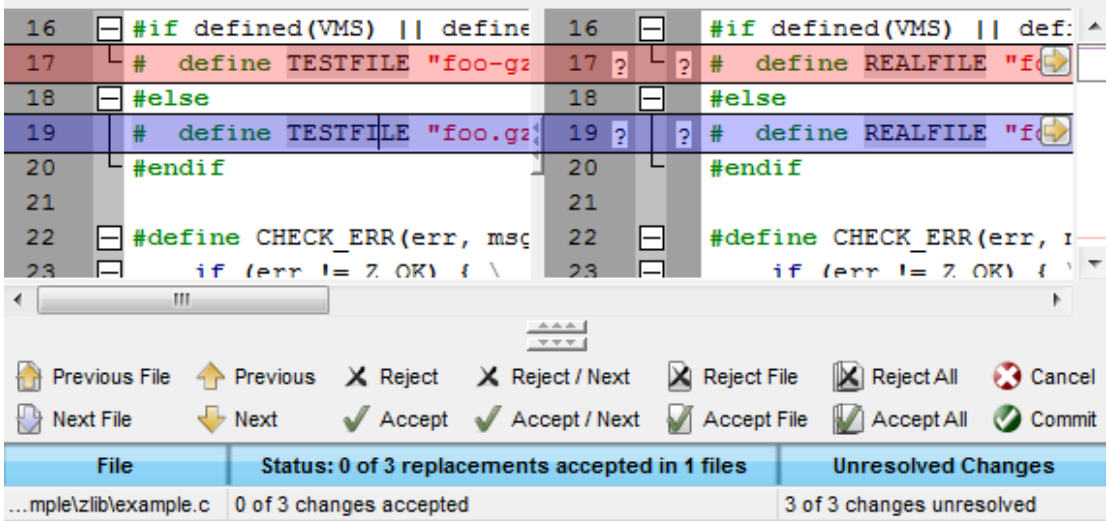
除了多文件查找，Understand 2.5 还提供了多文件替换操作，选择菜单栏 **Search>Replace in Files**，即可打开多文件替换对话框。



本对话框多数选项与多文件查找对话框中相应内容类似，不过多出了 Replace 输入域，用于输入期望替换查找对象的文本。

点击 **Preview Replace**，Understand 2.5 会对所有未保存的文件进行检查，如果存在未保存文件，在预览修改之前需要点击 Yes 以保存未保存的修改。

所有文件保存之后，弹出替换预览窗口。通过这个窗口可以单个替换，文件内替换或所有替换进行接受或者放弃操作。



替换预览窗口上方左侧显示的是替换前的代码，右侧显示的是替换后的代码，其中替换条目以红底标注，当前选中的替换条目以蓝底标注。左侧提供的 **Hide Common Lines** 选项可以隐藏不受替换影响的大部分代码。

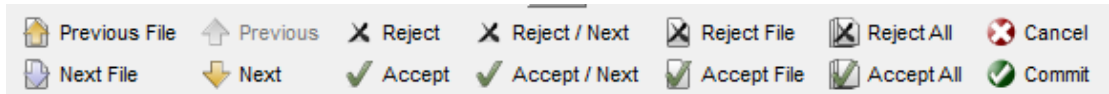
中间区域以 patch file 格式显示替换条目，patch file 支持 Unix patch 工具及其他类似程序操作。本区域可以通过点击上方的  隐藏。

下方区域列出替换所在文件，接受和未接受的替换条目。

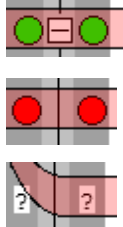
导航按钮可以控制文件和替换的跳转。接受按钮和拒绝按钮可以分别针对单个替换，文件



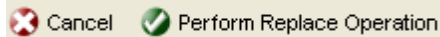
内替换或所有替换进行接受或者放弃操作。



接受的替换在代码显示区域以绿色圆圈表示，拒绝的替换以红色表示，没有确认过的替换以问号标识。点击红色或者绿色圆圈可以切换选择。



对所有的替换进行了确认以后，点击 **Perform Replace Operation**，弹出确认对话框，显示执行的替换数目。

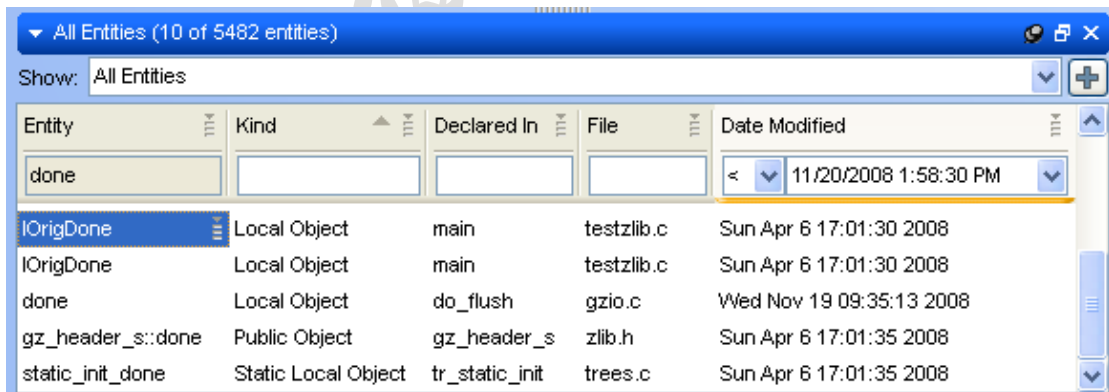


点击 **Cancel** 可以放弃此次修改。如果之前对某些替换做过接受操作，Understand 2.5 会弹出是否需要放弃修改的确认对话框。

## 实体定位器

实体过滤器的分类表页不能显示所有实体，通过实体定位器能够对任何实体进行查找和进一步了解，实体定位器提供了具有过滤功能的列表项对数据库中的所有实体进行检索，过滤选项包括名称，实体类型，实体声明的位置，实体所属容器以及实体的上次修改时间。此外，Understand 2.5 还支持通过层级结构对实体进行分类。

通过菜单栏上 **Search>Find Entity** 或者 **View>Entity Locator** 即可打开实体定位器。



与 Understand 2.5 的其他类型窗口一样，实体定位器也支持右键菜单。

## 列大小重设

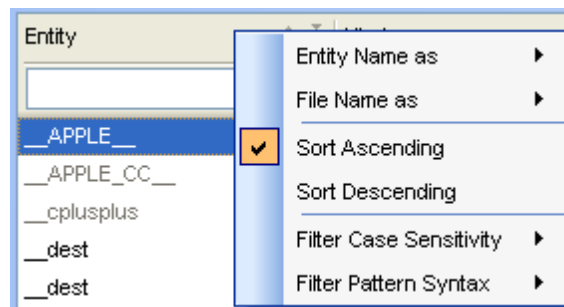
可以设置上图中各列宽度来控制显示的列数，拖动两列头之间的分割线即可修改其宽度。或者当双箭头符号出现时，双击列头分割线可以将某列进行最大化以查看该列中所有内容。

## 长名称和短名称

在实体定位器的 **Declared In** 和 **File** 列, 右击列头或者点击下拉图标可以指定实体名和文件名的显示方式。对于实体, 可以选择短名称或者全名 (包含编译单元名称); 对于文件, 可以选择短名称, 全名称或者相对路径名称。

## 列头

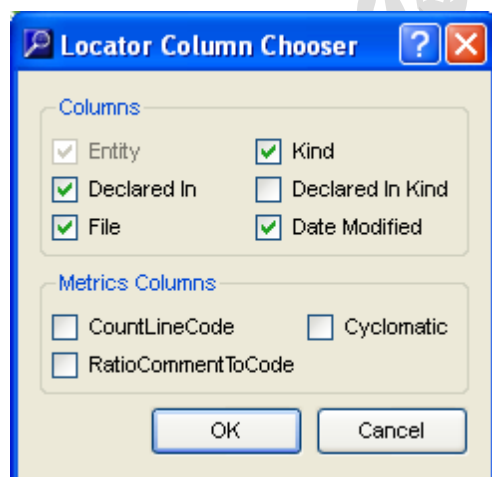
通过列头可以对实体定位器进行一些操作。左击可以对内容进行排序, 右击或者点击下拉图标可以控制实体的过滤, 排序和显示。



每列的实体都可以进行排序, 左击列头可以对排序方法进行切换, 默认的排序方法按照实体名称按照升序排列。

## 列选择

点击实体定位器右上方的+图标可以打开定位器的列选择器。



除了 **Entity** 必须选中, 其余各列可以根据需要进行选择。

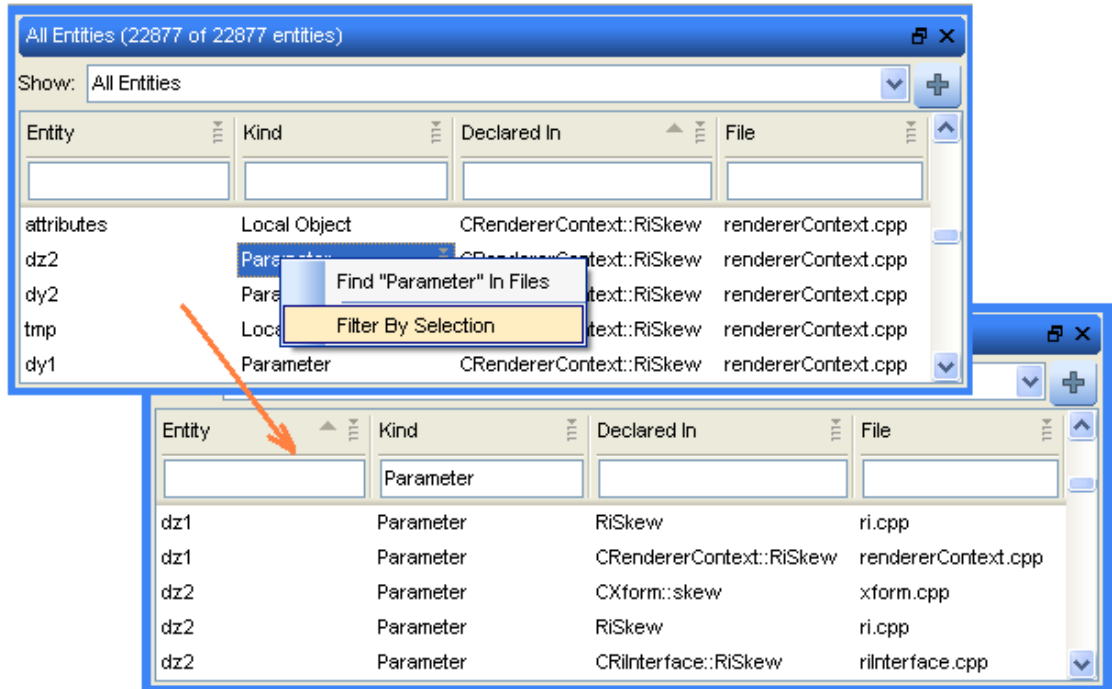
## 过滤

各列头下方的输入域可以对本列内容进行过滤, 输入域接受手动输入或者右键菜单的选

项输入。

例如，通过右击 Kind 列中某一条目，选择 Filter By Selection，即可根据选中的 Kind 内容进行过滤。这一操作使得实体定位器仅保留选中的 Kind 内容，标题栏则显示了此次过滤的匹配数目。或者，直接在输入域输入过滤内容也可进行过滤操作，删除内容可以清除过滤器。

以下例子展示了根据实体类型进行 Filter By Selection 操作：



对 Date Modified 列进行过滤时，可以通过左下拉菜单选择选择<, <=, >=, >操作符，然后从右下拉菜单的日历选择日期，时间可以通过输入进行修改。

相似的，Metrics 列允许通过比较运算符进行过滤。例如，可以通过过滤列出圈复杂度大于某个值的实体或者注释代码比小于某个值的实体。

右击某个列或者点击下拉图标可以打开该列上下文菜单，对大小写匹配和匹配模式进行设置：

- **Fixed String**：全字匹配。
- **Wildcard**：通配符匹配。
- **Regular Expression**：正则表达式匹配，正则表达式不支持大小写匹配选项。

下表列出正则表达式的一些功能字符：

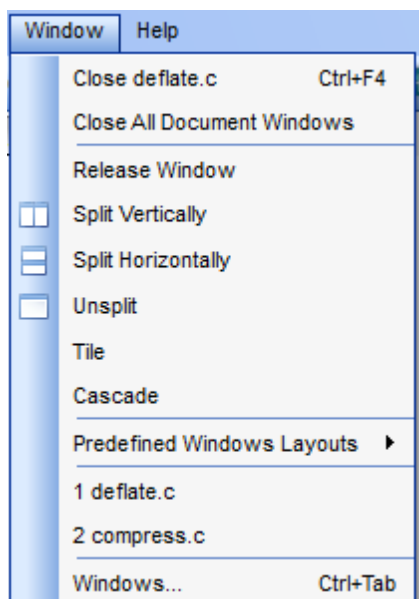
字符	描述	示例
^	匹配行首	^word 查找以 word 开头的所有行
\$	匹配行尾	word\$ 查找以 word 结尾（空格也计算在内）的所有行
\<	匹配单词开头	\<word 匹配以“word”开头的单词，如“wordless”和“wordly”，不匹配“fullword”和“awordinthemmiddle”
\>	匹配单词末尾	\>word 匹配以“word”结尾的单词，如“keyword”和“sword”，不匹配“wordless”和“awordinthemmiddle”
.	单字匹配的行查找	w.rd 查找包含 word, ward, w3rd, forword 等内容的行
*	对*号前的字符出现次数不做限制	word* 匹配 word, wor, work 等

+	+前的字符需出现至少一次	word+d 匹配 word, worrd, worrrd 等
?	? 前的字符可出现一次或不出现	word 匹配 word, wod
[]	匹配括号内字符	[AZ]查找包含 A 或者 Z 的所有行。 [Kk][eE][Nn] 查找 “Ken”, “KEen” 或者 “KeN”
[^]	匹配括号外字符	[^AZ]查找不包含 A 或 Z 的所有行
[-]	匹配范围指定	[A..Z]查找包含 A 到 Z 中任意字符的所有行, 不包含小写
	表达式的“或”关系	word leter 匹配 word, leter, letter, letter 等
\	将一个正则表达式功能字符转换成普通字符	\*/\$可以查找*, 本例表示查找所有以*/结尾的行

本文档不对正则表达式进行详细介绍。UNIX 用户可以使用“`man -k regex`”查看正则表达式使用说明。对于希望深入理解正则表达式的用户，可以参考 O'Reily 出版的《精通正则式》。

## 查找窗口

通常情况下，用户会打开多个窗口，通过菜单栏上的 **Window** 和 **View** 可以对这些窗口进行组织和针对性查找。



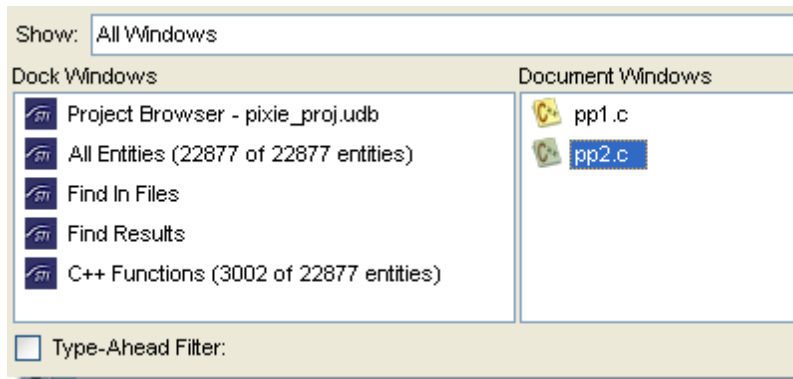
通过 **Window>Close** <current\_window> 关闭当前打开的文档窗口，**Window>Close All Document Windows** 关闭所有代码文件，图形视图及其他文档窗口，在当前使用的窗口的标签页右击，然后选择 **Close All But This**，**Close All Tabs to the Left** 或者 **Close All Tabs to the Right**。

选择 **Window>Release Window**，固定在标签页区域的窗口被分离成独立的窗口，可以自由移动。

**Window** 目录还提供 **Window>Split Vertially** 和 **Window>Split Horizontally** 将文档区域分割成多个区域。分割文档区域时，文档区域被根据选项分割成两个区域，可以将一边的标签页拖动到另一边。选择 **Window>Unsplit** 可以取消分割。**Window>Tile** 和 **Window>Cascade** 也可以对打开的窗口进行其他风格的排列。

**Window>Predefined Window Layout** 可以选择若干标准布局，选项包括“**Tight**”，“**Classic**”和“**Multi-monitor**”。

通过 **Window>Windows** 或者 **Ctrl+Tab** 打开当前打开窗口的临时列表，双击该列表中的的某一项，该列表被关闭，选中窗口在文档区域被打开。按 **Esc** 可以直接关闭列表。

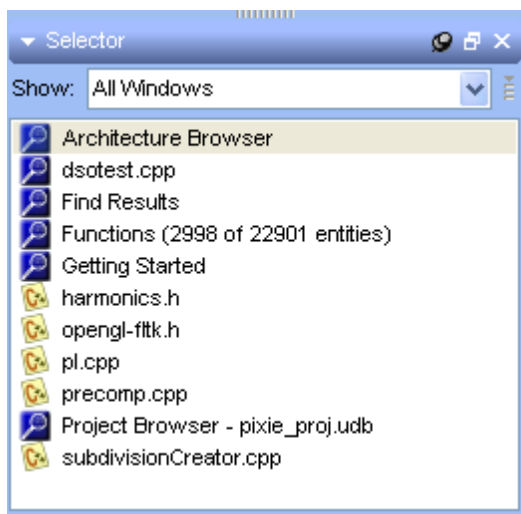



通过在本区域的 **Show** 列表选择窗口类型可以减少列出的窗口数目。勾选 **Type-Ahead Filter**，输入期望查找的窗口的名称的字符，也可以快速定位需要查找的窗口。

## View 菜单

通过 **View>Last Parse Log** 可以打开之前得到的工程分析日志。

**View>Window Selector** 显示当前打开的窗口的一个列表，点击窗口名字可以激活该窗口。默认情况下，该选择器显示所有窗口，用户也可以指定选择编辑器或者其他类型的窗口。窗口名称前的图标显示了窗口类型，还能显示源代码文件的修改状态。



通过下拉图标  可以选择排序方法，点击一个条目使对应窗口获取焦点。

当选择器区域处于激活状态，通过底部的过滤器可以快速缩窄显示范围。

通过窗口选择器可以方便的进行很多操作，如在选择器选中多个窗口后右击，选择 **Close Selected Window(s)** 或者 **Close Unselected Window(s)** 可以对多个窗口进行关闭操作。

如果在代码中创建书签，菜单栏中 **View>Bookmarks** 可以打开显示当前所有书签的列表。

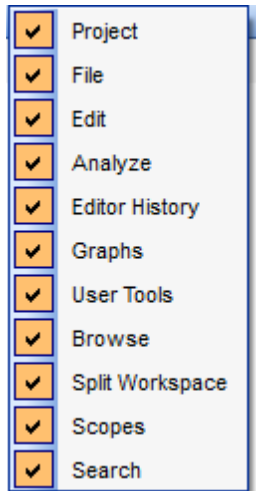
## 工具栏显示

右击菜单栏或者工具栏对其分类进行选择即可控制相应内容的显示。工具栏被分成以下分

---

类: 工程, 文件, 编辑, 分析, 编辑器历史, 视图, 用户工具, 浏览, 工作空间分割, 范围和查找。

使用菜单栏 **View>Toolbars** 也可以实现上述操作。



---

## 代码编辑

本章介绍 Understand 2.5 的代码及文本文件的编辑器。

本章包含以下内容

代码编辑器

代码保存

代码查找

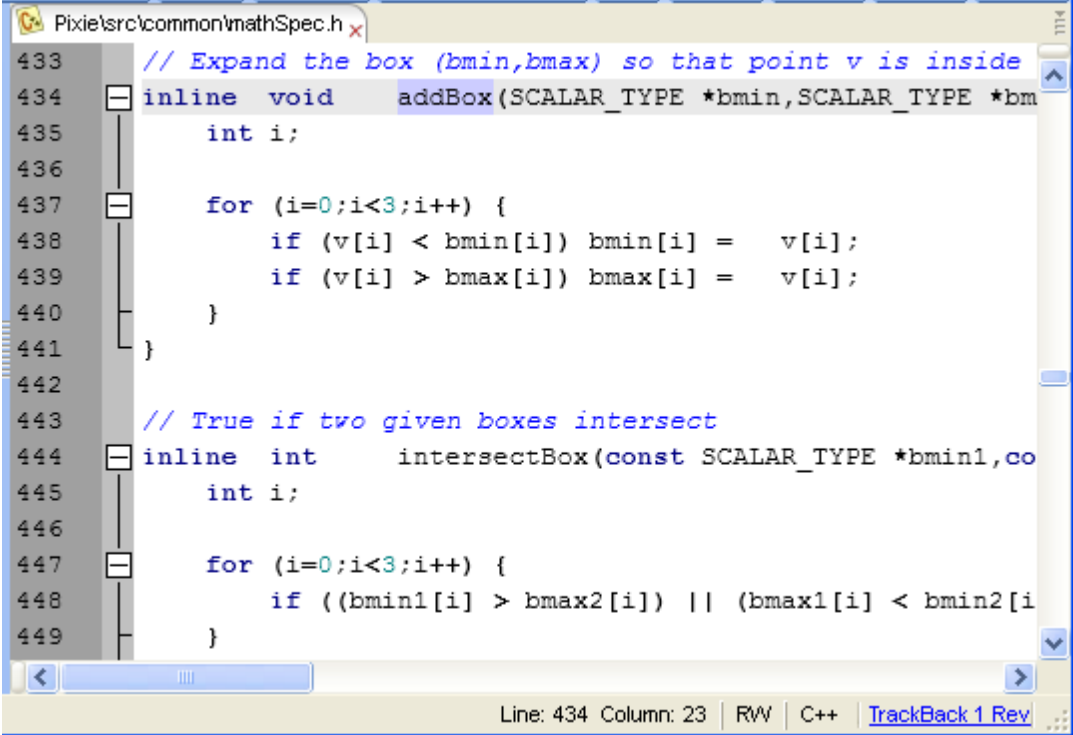
其他特性

代码打印视图

Shiningstone provided

## 代码编辑器

Understand 2.5 的代码编辑器提供了全特性的代码编辑功能，还包括语法着色功能和右键激活查看实体信息功能。



```
433 // Expand the box (bmin,bmax) so that point v is inside
434 inline void addBox(SCALAR_TYPE *bmin,SCALAR_TYPE *bm
435     int i;
436
437     for (i=0;i<3;i++) {
438         if (v[i] < bmin[i]) bmin[i] = v[i];
439         if (v[i] > bmax[i]) bmax[i] = v[i];
440     }
441 }
442
443 // True if two given boxes intersect
444 inline int intersectBox(const SCALAR_TYPE *bmin1,co
445     int i;
446
447     for (i=0;i<3;i++) {
448         if ((bmin1[i] > bmax2[i]) || (bmax1[i] < bmin2[i
449     }
```

Line: 434 Column: 23 | RW | C++ | [TrackBack 1 Rev](#)

通过 **Tools>Option** 打开选项对话框（参见 [Editor](#)），在 Editor 类别配置可以控制行号和用于控制代码块的展开和收缩的扩展标记的显示。除此之外，显示字体和其他一些特性都可以根据需要进行设置，书签使能，缩进标识，右边界标识也可以在这个类别设置。

通过菜单栏 **View>Zoom** 可以对文本显示进行放大缩小。

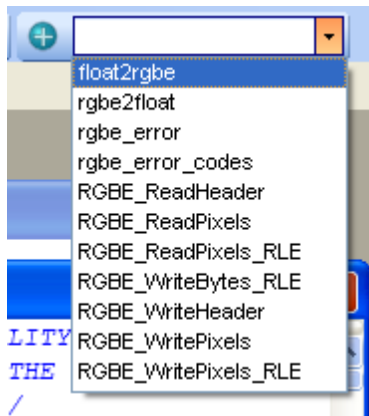
Understand 选项对话框的 **Editor>Styles** 类别（参见 [Editor>Styles](#)）可以对不同功能类型代码的颜色进行设置。**Key Binding** 类别（参见 [Key Binding](#)）显示了编辑器可以显示组合功能键。

## 审视列表

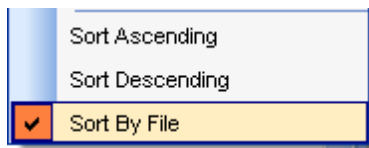
从工具栏上的审视下拉列表选中当前文件的函数，过程及其他语言定义结构可以跳转到对应位置，该下拉列表显示了上次工程分析得到的类似结构。

点击+图标将该列表移到与实体过滤器相同区域的审视标签页，显示当前代码文件涉及的相关结构体，用于在大型文件中快速定位。





审视列表区域列表的右键菜单提供了排序方法选择，包括 Sort Ascending 和 Sort Descending 及默认选项 Sort By File。



选择 **View>Scope List**，可以打开审视列表，显示出当前文件包含的函数，过程及语言相关的结构体，名称旁边的数字指示文件中实体声明位置的行号。单击其中的某个条目可以在信息浏览器中浏览相关信息，双击则可以跳转到对应实体声明或创建的位置，并将当前文件中对应内容高亮显示。

上下文信息侧边栏（参见）提供了与视野列表类似但更为强大的功能。

## 状态图标

代码编辑器的每个文件在左上方的标题栏显示状态图标，图标中的字母标识了文件类型，其颜色显示了文件的分析状态。文件名旁边的星号标识了该文件修改后未被保存。



分析过的工程文件（未被修改）



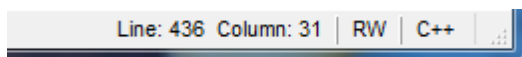
修改过的工程文件（需要被分析）



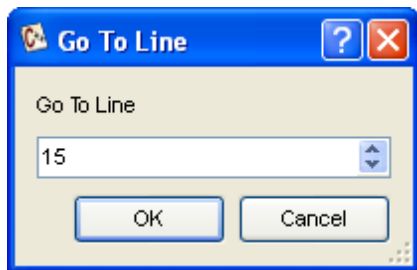
非工程文件

## 状态栏

当代码编辑窗口被激活，窗口底部的状态栏显示当前的行号和列号，文件读写权限以及代码语言类型。



点击状态栏中的行号在菜单栏选择 **Search>Go to Line**，可以打开行号选择对话框。



点击 RW 指示，读写模式会切换到 RO。

点击语言指示，可以选择 Understand 2.5 对当前文件的语言类型处理方式。

## 文本选择和拷贝

Understand .2.5 窗口区域的大部分文本可以按照所在操作系统的操作方式被拷贝或者剪切到 Windows（或 X11）系统的剪贴板。对于 Windows 系统，按下鼠标左键然后拖动，或者按下 Shift 键通过箭头键或者鼠标移动光标即可选取文本。通过菜单栏 **Edit>Select All** 或者使用右键菜单可以快速选取整个文件。

按下 Alt 键（在 X Windows 系统中的 Ctrl 键），可以选择代码中的一片矩形区域——例如，移除拷贝文本左边缘的 tab 字符。这种矩形区域内容也可以向代码区域粘贴。

选中文本后，通过菜单栏的 **Edit** 菜单或者右键菜单，使用 **Cut** 或者 **Copy**，然后就可以将选中内容粘贴到其他应用程序。

## 浏览模式

通过点击主工具栏的 **Browse** 按钮或者菜单栏中 **View>Browse Mode**，可以将代码编辑器切换到“浏览”模式，处于“浏览”模式时，**Browse** 按钮呈高亮显示。



浏览模式下，代码中的各种实体都像链接一样处理。鼠标移动到一个实体，下划线显示在链接下方。点击该链接，即可跳转到对应实体的声明位置，信息浏览器内容也随之更新。

如果找不到对应实体的声明，状态栏会显示相关信息，计算机也会发出蜂鸣告警。

浏览模式下，依然支持文件编辑和键盘、鼠标右键的操作，只有鼠标左键操作有所不同。

代码编辑窗口激活的状态下，按下 **Ctrl** 键可以临时进行浏览模式的操作，通过 **Ctrl+Alt+B** 可以切换浏览模式。

浏览模式下的控制行为参见 [Edit>Browse](#)。

## 右键菜单

代码编辑窗口的右键菜单提供了很多浏览编辑操作的快捷方式，也提供了相关实体的特定信息。

以下介绍了右键菜单包含的典型的浏览操作（与点击对象有关）：

- **View Information** 信息查看（参见[信息浏览器](#)）
- **Graphic Views** 图形视图（参见）
- **Edit Source/Definition** 代码编辑（参见）

- 
- User Tools 用户工具（参见）
  - Explore 浏览（参见[层级查看](#)）
  - Find In... 查找（参见[多文件查找](#)）
  - Add Favorite 添加到收藏夹（参见[收藏夹](#)）
  - Metrics Chart 度量图表（参见）

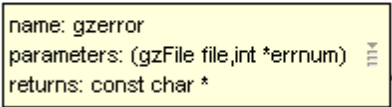
以下介绍了右键菜单包含的典型的编辑操作（与点击对象有关）：

- Undo/Redo 撤销/重做
- Cut/Copy/Paste 剪切/复制/粘贴（参见[文本选择和拷贝](#)）
- Select All 全选（参见[文本选择和拷贝](#)）
- Jump to Marching Brace 跳到块末
- Select Block 块选择
- Hide/Show Inactive Lines 行显示/隐藏
- Fold All 折叠所有分支
- Soft Wrap 软包装
- Comment Selection/Uncomment Selection 注释选择/代码选择
- Change Case 大小写切换
- Revert 反转
- Add Bookmark 添加书签

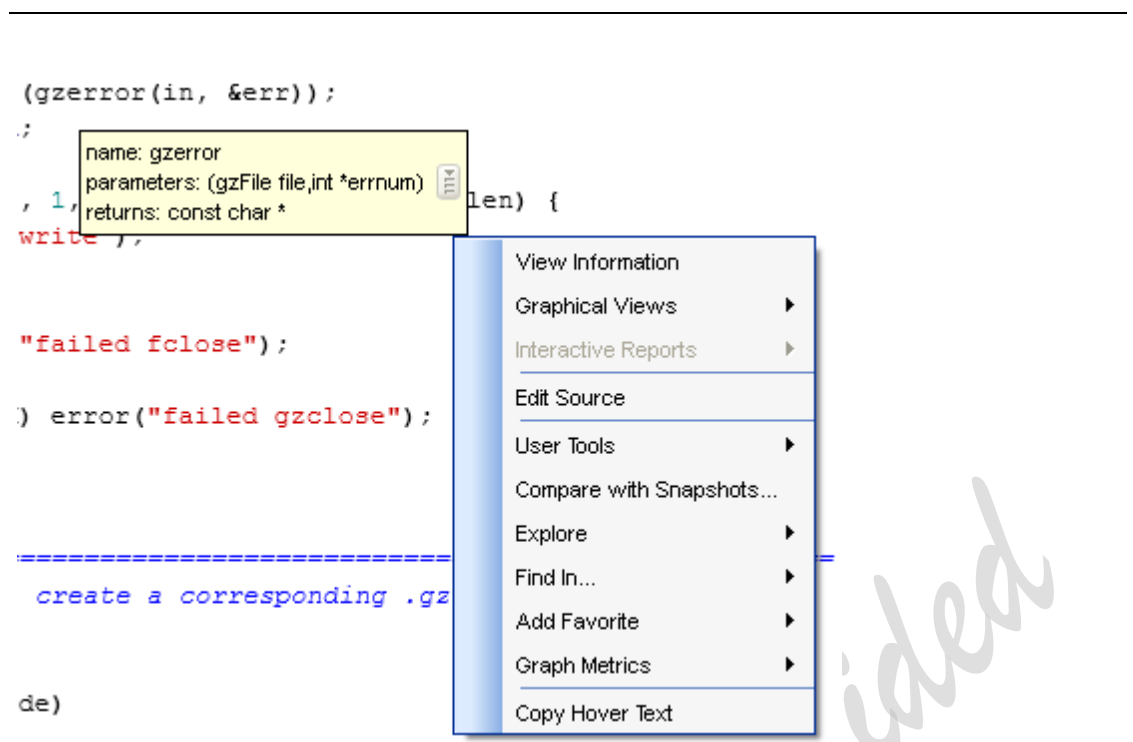
## 活动菜单

将鼠标光标移动到代码中的任意实体，会显示一个消息框，内含该实体的声明信息。例如，指向变量会显示出变量类型，指向常量会显示出常量值，指向一个函数调用会显示出该函数的参数和返回值。

```
if (len < 0) error (gzerror(in, &err));  
if (len == 0) break;  
  
if ((int)fwrite(buf, 1,   
    error("failed fwrite");
```



点击消息框上的下拉按钮（上图中圆圈标示），会随即弹出活动菜单，该菜单包含了一些相关实体右键菜单的功能选项。活动菜单额外还包含了 **Copy Hover Text** 选项，该选项提供对消息框内的内容的拷贝操作。



## 代码保存

完成代码编辑后，点击，或者输入 **Ctrl+S**，或者通过 **File>Save** 对修改进行保存。

通过 **File>Save As** 可以将当前文件保存到另一个不同文件名的文件，如果操作文件属于工程文件，会弹出对话框询问是否需要将该文件添加到工程。

如果需要对多个文件进行保存，点击，或者选择 **File>Save All**。

如果不希望对上次保存后所做的修改进行保存，可以右击文件选择 **Revert**。

通过菜单栏上 **Window>Close <current file>**可以关闭当前文件，**Window>Close All Document Windows** 可以关闭所有文件。文件标签的右键菜单，还提供了 **Close**, **Close All But This** 和 **Close All Tabs to the right** 选项。

## 代码查找

Understand 2.5 提供了多种方法对代码中的字符串进行查找，以及行定位。相关命令都收纳在 **Search** 菜单项，下表列出了介绍各种方法的相关章节：

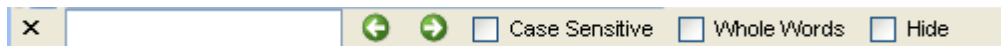
查找命令	相关章节
实体定位器	
多文件查找	
多文件替换	
快速查找	
查找匹配浏览	
查找和替换	
历史	
括号匹配	
收藏夹	

上下文信息边栏	
书签	

Understand 2.5 代码浏览功能的更详细介绍参见[窗口介绍](#)。

## 查找匹配浏览

使用 **Ctrl+F**（或者通过 **Search>Find**）可以在文件中快速启动查找，此时编辑窗口的状态栏自动变成查找栏。

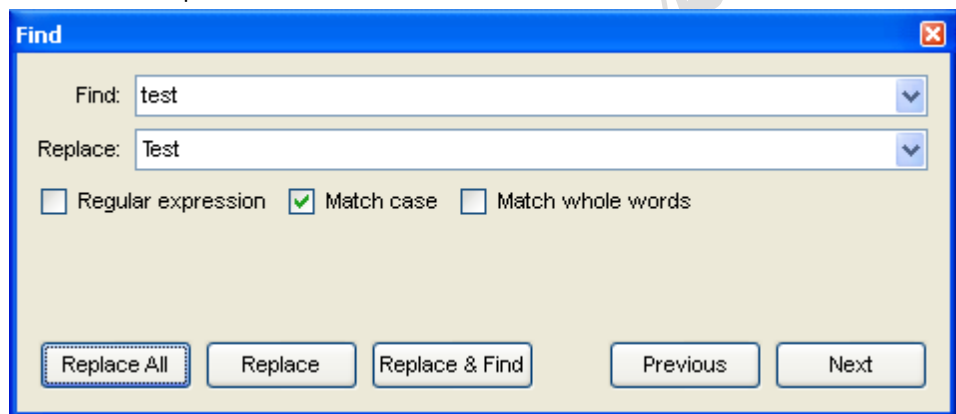


在输入域输入一个字符串，编辑器中匹配文本随即高亮显示。点击前进或者后退图标，可以在各个匹配项之间逐个切换。此外，还可以通过 **Case Sensitive** 和 **Whole Words** 选项对查找方式进行设置。

勾选 **Hide** 的情况下，点击代码会将查找栏隐藏，使用 **Ctrl+F** 可以恢复查找栏的显示。**Ctrl+Shift+F** 对前一匹配项进行查找。

## 查找和替换

使用查找对话框，可以进行查找替换操作和正则表达式查找，该对话框通过菜单栏的 **Search>Find&Replace** 或者 **Ctrl+Alt+F** 打开。



在 **Find** 输入域，输入期望查找的字符串。

使用 **Regular expression**，**Match case** 和 **Match whole words** 选项控制查找方式。通过 **Regular expression**，可以使用 **UNIX** 风格的模式查找。正则表达式的部分功能介绍参见[列表过滤](#)。

在 **Replace** 输入域，输入期望替代原字符串的新内容。

使用 **Previous** 和 **Next** 控制查找方向，找到匹配项时，点击 **Replace All**，**Replace** 或者 **Replace&Find** 对当前匹配项进行操作。

查找对话框仅针对单个文件进行查找，如果需要对多文件进行查找，参见[多文件查找](#)。

## 代码浏览记录

使用工具栏上的前进、后退图标可以在代码浏览点的历史记录之间进行切换。




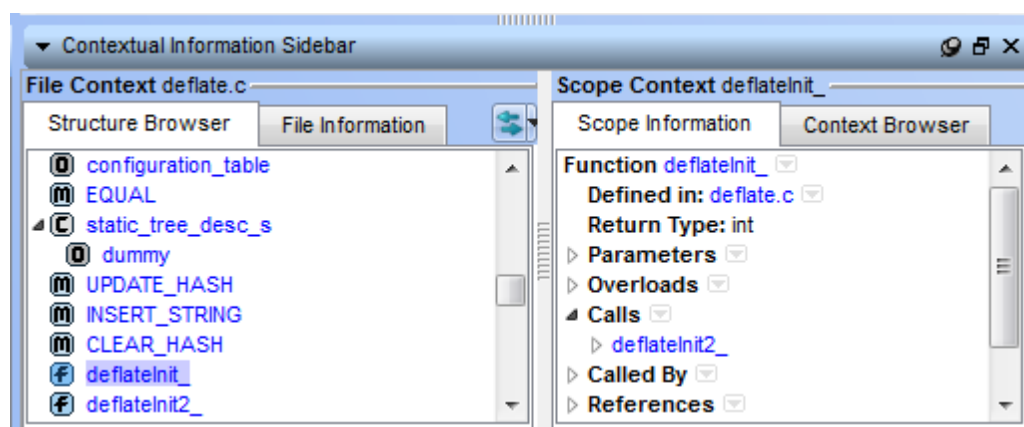
下拉箭头可以显示浏览记录的所有条目。

浏览记录以行号存储，如果希望保存实体的记录，参见[收藏夹](#)。

## 上下文信息边栏


上下文信息边栏（CIS）功能与范围列表（参见范围列表）类似，但是更为强大。通过目录

栏 **View>Contextual Information** 或者工具栏上的  图标可以打开上下文信息边栏。



CIS 显示当前激活的编辑窗口的结构和信息，标签页包含了以下信息：

- **Structure Browser:** 提供当前文件的扩展范围列表，列出了文件中函数，头文件包含，宏定义，类及其他各种结构的名称。名称旁边的图标指示了实体类型。将鼠标移动到一个条目，弹出显示实体类型和名称的活动文本框。使用 **Ctrl+F** 可以在标签页范围内进行查找。
- **File Information:** 提供当前文件的信息浏览器。
- **Scope Information:** 显示 **Structure Browser** 中选中实体的相关信息。
- **Context Browser:** 在左侧显示当前实体在层级结构中的位置，右侧显示当前实体包含的其他实体。

文件信息标签页右上角的  图标（或者使用 **Ctrl+,**）可以将编辑窗口和 CIS 中的文件替换为同目录下同名但不同扩展名的文件。例如，使用这种方法可以在 .p/.cpp 和 .h 文件之间快速切换。

右键菜单也提供了多种快捷的操作选择。

## 其他特性

代码编辑器还提供了其他一些文件显示和编辑的操作。

## 括号匹配

括号匹配是 **Understand 2.5** 提供的一项快捷的语法相关的操作，能够快速查找到当前括号对应的闭括号，如 **()**，**{}** 和 **[]**，这项查找忽略注释区域。

当鼠标位于能够识别的符号处，使用 **Ctrl+j**（或者在右键菜单中选择 **Jump to Matching Brace**）即可在括号之间快速跳转，再次输入可以返回原处。使用菜单栏上的 **Search>Go to**

**Matching Brace** 也能够达到同样效果。

使用 **Ctrl+Shift+J**（或者在右键菜单中选择 **Select Block**）可以选中括号之间的文本。

对于不能匹配的括号以红色高亮显示，匹配的括号以绿色高亮显示。

该项操作还支持预编译语句（例如 **#ifdef** 和 **#endif**），操作方法为输入 **Ctrl+j**，或者在右键菜单中选择 **Jump to Matching Directive**。

折叠和隐藏

使用行号旁边的+、-可以将代码中的函数，if 语句及其他一些有语法起始和结束的代码行进行“折叠”。

```
6  #define __config__
7  - #ifndef std_include
8  + #ifndef nested
15 L #endif
```

在代码区域单击右键，选择 **Fold All** 可以将所有代码块进行折叠，菜单栏上的 **View>Fold All** 也可以实现此项功能。

使用 **Hide inactive Lines** 可以将通过宏控制的条件编译的无效代码隐藏，**Show Inactive Lines** 可以使这些代码重新显示。菜单栏上的 **View>Hide Inactive Lines** 也可以实现此项功能。

注释与取消注释

对选中代码的右键菜单选择 **Comment Selection**，可以将其注释，**Uncomment Selection** 则是取消注释。菜单栏上的 **Edit>Comment Selection** 和 **Edit>Uncomment Selection** 也可以实现此项功能。

大小写修改

Understand 2.5 代码编辑器支持选中文本进行快速的大小写修改，方法如下：

- 1 选中代码中的一个或者多个单词。
- 2 通过菜单栏 **Edit>Change Case**，或者右键菜单中选择 **Change Case**。
- 3 选择需要使用的大小写修改方式，选项包括：

选项	快捷键	原始文本	结果
LowerCase	Ctrl+U	Test_me	test_me
UpperCase	Ctrl+Shift+U	Test_me	TEST_ME
Invert Case	Ctrl+Shit+l	Test_me	tEST_ME
Capitalize	Ctrl+Alt+U	Test_me	Test_Me

行包装

通常情况下，当编辑窗口的宽度不足以显示整行内容时，行内容被截断。Understand 2.5 通过编辑窗口的右键菜单提供的 **Soft Wrap** 选项，支持将整行的文本以多行的方式显示出来。菜单栏 **View>Soft Wrap** 也可以实现此项功能。

这种断行仅限于显示，而不会影响实际代码。



---

代码打印的包装模式设置参见 [Editor>Advanced](#)。

## 键盘命令

通过 **Tools>Options** 选择 **Key Binding** 分类，可以查看代码编辑器支持的组合键列表。例如，**Ctrl+Alt+K** 将从光标位置到行尾的内容进行剪切，**Ctrl+T** 将光标所在行与上一行进行交换。

使用 **Help>Key Bindings** 也可以打开组合键列表，查找“Editor”（110 行）可以跳转到编辑窗口组合键信息的开始部分。

## 宏指令记录和使用

Understand 2.5 支持将期望重复执行的编辑操作记录成宏指令，在随后的时间使用该指令替换一系列操作。

宏指令记录参照如下步骤：

- 1 从菜单栏选择 **Tools>Editor Macros>Record Macros**，或者直接使用 **Ctrl+Alt+M**。
- 2 执行期望记录的操作。
- 3 再次执行 1 中的操作。

将光标移动到期望执行操作的区域，通过菜单栏的 **Tools>Editor Macros>Replay Macro**，或者输入 **Ctrl+M**，就可以将记录的宏指令对选中区域进行操作。

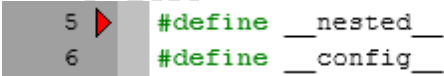
## 文件创建和打开

在编辑窗口从菜单栏选取 **File>New>File**，可以创建一个未命名的空白文件。通过 **File>Open>File**，可以打开工程中或者工程之外的文件。

右击文件名，上下文菜单提供了 **Edit File** 和 **Edit Companion File**，如对于 `encrypt.c`，其伙伴文件为 `encrypt.h`。

## 书签

在代码区域选中一行单击右键，在菜单中选择 **Add Bookmark** 即可创建书签，菜单栏 **Edit>Bookmarks>Toggle Bookmark** 也可以实现此项功能，设置书签的行首由红色箭头标识。

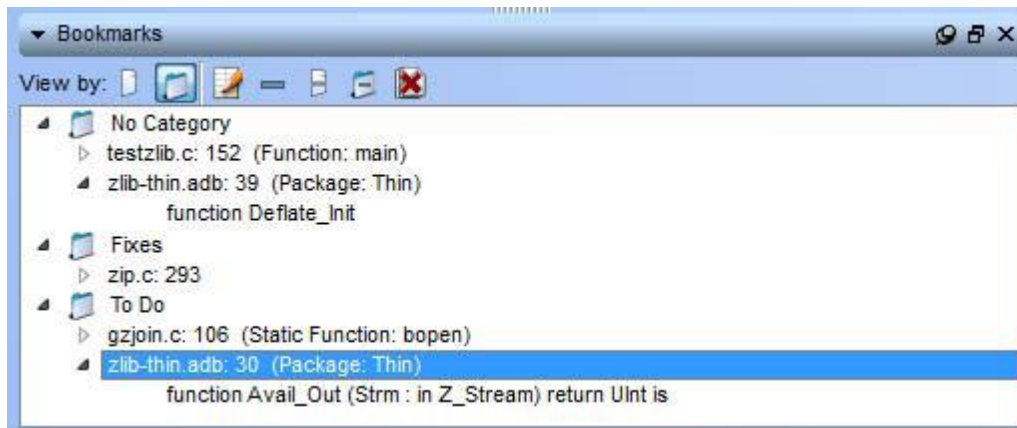


```
5  #define __nested__
6  #define __config__
```

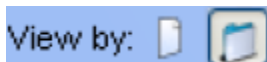
对于包含多个书签的文件，使用右键菜单中的 **Previous Bookmark** 或者 **Next Bookmark**，可以在书签位置之间快速跳转。菜单栏 **Edit>Bookmarks** 也提供了相同功能的选项。

通过菜单栏 **View>Bookmarks** 可以打开包含所有文件中的书签的窗口，双击其中的书签即可跳转到代码中的对应位置。如果书签创建在一个实体内部，书签窗口会将包含该书签的实体名称和类型也显示出来。例如，在某函数首行创建书签，书签窗口会显示这个函数的名称。





书签窗口提供了工具栏对书签进行如下管理操作：



用于切换文件视图和分类视图，文件视图中展开文件可以查看该文件中所有书签，分类视图按照用户对书签的分类进行显示。



用于对书签进行归类，选中书签，在弹出列表中选择类别即可实现。输入名称点击 OK 直接创建新类别。



用于删除选中书签。



用于在文件视图删除选中文件中的所有书签。选中书签时点击此按钮也会将同文件中所有书签删除。



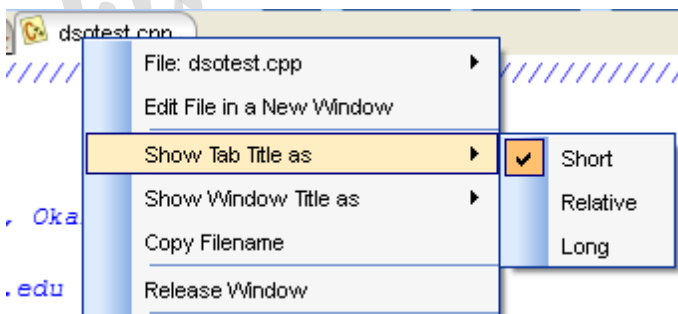
用于在分类视图删除选中分类中的所有书签（分类本身不会被删除）。




用于删除所有书签。

## 编辑器标签页管理

右击编辑器上方的标签页，可以对标签页的行为进行配置。



通过 **Show Tab Title As**，可以设置标签页对应文件名的显示长短，同样，通过 **Show Window Title As** 修改 Understand 2.5 窗口及其他编辑器窗口标题栏的显示长短。

使用 **Release Window** 将标签页变成一个可以随处放置的独立窗口，点击  将一个标签页变成 Understand 2.5 窗口中的一个子窗口。

## 修改代码字体大小

通过 **Tools>Options** 打开选项对话框，在 **Editor** 分类中可以设置默认的显示字体和字体大小（参见 [Editor](#)）。

此外，通过 **View>Zoom** 可以针对单个代码窗口进行字体大小显示的调整。**View>Zoom>Zoom In** 放大字体，**View>Zoom>Zoom Out** 缩小字体，**View>Zoom>Reset Zoom** 将字体大小恢复成默认大小。

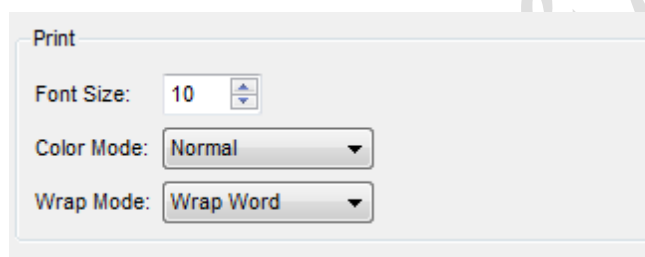
## 字母序行排列

对话框选项框的 **Key Binding** 分类页面提供了排序方法选择的操作，设置组合键实现该操作的方法参见 [Key Bindings](#)，然后通过组合键对选中多行代码进行字母序排列。

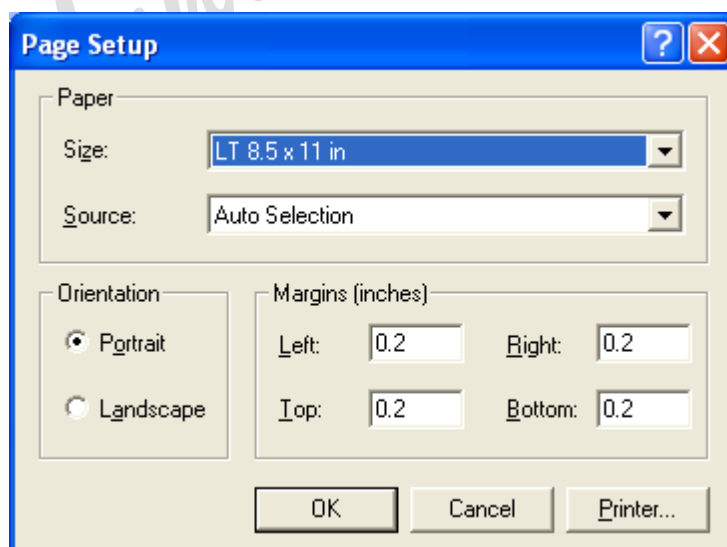
## 代码打印视图

使用菜单栏 **File>Print** 可以打开标准打印对话框，对当前显示的代码进行打印，打印页面设置为每页 66 行。

默认情况下，文件按照屏幕上显示的字体和颜色进行打印，也可以通过选项对话框对代码打印进行定制。选择 **Tools>Option** 打开选项对话框，选择 **Editor** 分类页面的 **Advanced** 分类，具体设置参见 [Advanced](#)。



通过菜单栏的 **File>Page Setup** 可以在不改变屏幕显示的情况下修改打印输出，对话框提供类似下图的选项（因操作系统略有不同）。



---

## 代码库层级结构化

本章介绍 Understand 2.5 提供的层级特性，及如何使用这些特性来分析代码。

本章包含以下内容

- 层级结构介绍
- 层级结构浏览器使用
- 层级结构依赖图
- 层级结构度量
- 层级结构管理
- 创建层级结构
- 构造层级结构
- 使用 XML 管理层级结构

Shiningstone provided

## 层级结构介绍

Understand 2.5 中的层级结构体现了代码的抽象层次。例如，一个职员层级结构包含一个特定项目中所有工程师的对应节点，每个工程师对应的节点包含该工程师所有或者有权修改的源代码文件，之后可以通过该层级结构衍生出相关的依赖关系和互动关系。

层级结构创建了代码单元（实体）的层次体系，通过层级结构可以对软件工程的不同区域以及软件层级的查找路径命名，用户可以选择使用系统提供的层级结构或者自定义的层级结构。

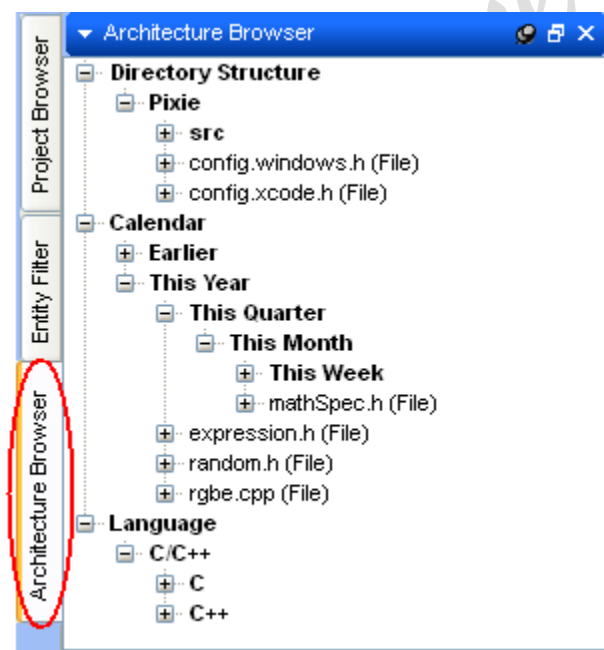
层级结构自定义相关实体的子集，因而不需对数据库的代码实体进行引用，并且可以多次包含一个特定实体（从技术上讲，层级结构的扁平化扩展不对这些实体的所有权进行维护）。

将层级结构相继联结可以创建出独特的实体过滤器。

从更技术性的角度，层级结构的组合和构成采用单集运算，过滤器产生的结果是一系列实体，这些实体可以看成是扁平化的，或者来自另一个层级结构。过滤器的定义可以保存为一个动态的层级结构，当数据库内容发生变化，该层级结构也会随之更新，在随后的某个时间重新设置过滤器。

### 层级结构浏览器使用

使用菜单栏 **Project>Architectures>Browse Architectures** 打开层级结构浏览器，该浏览器以可扩展的列表形式显示工程当前定义的层级结构。



层级结构区域与过滤区域类似，选中一个条目，信息浏览器内容自动更新为该条目的相关信息（信息浏览器的“Sync”需选中）。

### 浏览层级结构

使用“+”对所选条目进行扩展以便对层级结构进行深层次浏览，实体如文件、函数、变量

均显示在层级结构中。

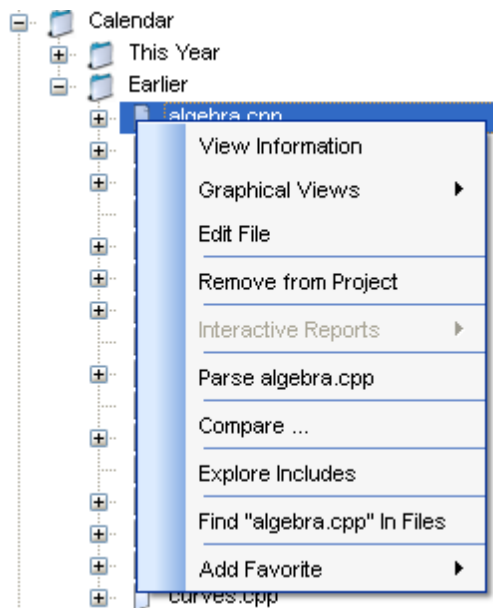
Understand 2.5 从以下范畴提供了“自动层级化”的操作：

- **Directory Structure:** 以文件系统的层级结构（目录及子目录）显示工程文件。
- **Calendar:** 根据上一次修改时间显示工程文件，修改时间范围节点包含本年度，本季度，本月，本周，昨天和今天。
- **Language:** 按照文件使用的语言和所在目录结构的位置显示工程文件（本层级结构视图只对使用多种语言的工程有效）。

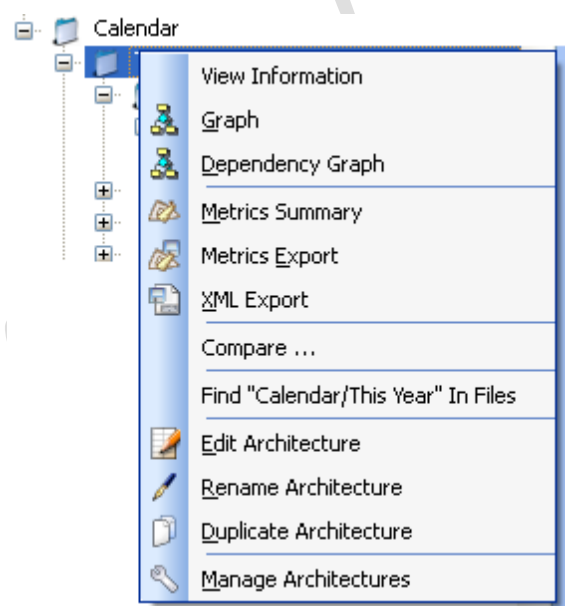
这些视图只有当工程被重新分析后才会进行更新，所以当代码发生了修改而没有对工程进行分析操作，这些视图（尤其是日历视图）就会过期。

层级结构浏览器也同样支持右键菜单，以快速向用户提供一些相关信息和操作。

层级结构中文件条目的右键菜单



层级结构中节点的右键菜单



层级结构节点（如文件系统目录节点和本季度修改节点）的右键菜单包含了以下一些其他对象的右键菜单没有提供的额外选项：

- **Graphical Views>Dependency Graphs:** 显示层级结构各节点的依赖关系，参见。
- **Metrics Summary:** 提供选中节点各实体的度量数据，这些度量数据基于当前节点下的实体，不包括子节点中的实体，参见。
- **Metrics Export:** 将度量总结导出到一个 CSV 文件，参见。
- **XML Export:** 将层级结构中选中节点及子节点包含的实体以 XML 格式导出，参见。
- **Edit Architecture:** 为用户自己创建的层级结构打开一个层级结构构建器，Understand 2.5 自动构建的层级结构不能被修改，参见。
- **Rename Architecture:** 为用户自己创建的层级结构或者节点打开一个层级结构重命名窗口供用户对其中名称进行修改，Understand 2.5 自动构建的层级结构不能被重命名，参见。
- **Duplicate Architecture:** 打开层级结构复制窗口对选中层级结构的副本进行命名，参见。
- **Manage Architecture:** 打开层级结构管理窗口，参见。

## 查看层级结构依赖关系图

Understand 2.5 支持将层级结构以图形形式显示，并可将其保存为 PNG, JPEG, SVG 以及 Visio 格式。

注意：类和包也支持依赖关系图。

关系图创建步骤如下：

1 选中期望绘图的顶层节点，绘图范围可以包含整个层级结构或者其中的一个子结构。

2 右击选中节点，选择 **Graphical Views**。根据选中对象不同，后续选项包括 **Graph Architecture**，**Depends On**，**Depended On By**，**Butterfly-Dependency Graph** 和 **Internal Dependencies**。如果选中的是层级结构节点，菜单栏中 **Graphs>Graphs for <current\_entity>**实现同样功能。

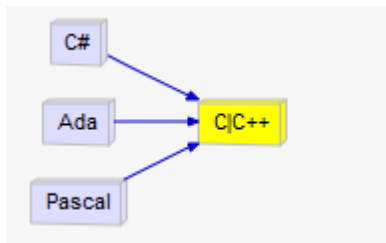
选择菜单栏 **Graphs>Dependency Graphs** 可以打开整个层级结构的内部依赖关系图。

Understand 2.5 为层级结构依赖关系图提供了与其他图相同的工具栏，具体操作参见。



将关系图保存到 JPG，PNG 或者 SVG 文件，参见。将关系图保存到 Visio 文件，参见。

除层级结构关系图之外的依赖关系图，Understand 2.5 提供了视图定制面板供用户对视图显示进行修改，包括扩展，高亮着色，基于节点箭头的控制，还支持撤销，重做以及保存和装载视图定制选项。例如，下图是多语言 zlib 示例工程中的 C/C++节点的默认 **Depended On By** 图。



各节点以 3D 盒子显示，双击可以将其展开，显示出包含的子节点，用户可以对这些节点一直执行展开操作直到到达文件层次。

<http://www.scitools.com/features/dependenciesGraphs.php> 包含了介绍如何使用视图定制的视频。

## 视图定制工具栏



视图定制工具栏提供以下操作按钮：

- **保存按钮**：弹出窗口提示用户输入保存当前设置的名称，保存对象仅限于视图的根节点和特定图形类型。如果存在保存过的设置，通过下拉菜单可以装载保存的设置，否则，需要为当前设置输入一个名称，然后点击 **Save**。

- **装载按钮**：弹出窗口供用户选择期望在当前窗口使用的图形设置，选项内容由之前保存过的设置确定，通过 **Graphs>Dependency Graphs>Load Saved Dependency Graph**，可以查看所有保存过的设置。

- **撤销按钮**：取消前一次修改。

- **重做按钮**：重新执行前一次取消的修改。

## 视图定制域

视图定制面板工具栏下面第一组选项对视图中所有节点生效，其中包括：

- **Hide Unhighlighted Edges**：仅对内部依赖视图有效，而且仅在节点连接关系被高亮显示的情况下才能使用。勾选此选项的情况下，Understand 2.5 将非高亮显示的箭头隐藏，并且忽略这些关系对视图重新组织。

• **Hide Nodes with No Highlightened Edges:** 仅对内部依赖视图有效，而且仅在节点连接关系被高亮显示的情况下才能使用。勾选此选项的情况下，Understand 2.5 将非高亮显示的箭头相关的节点隐藏，并且忽略这些节点对视图重新组织。。

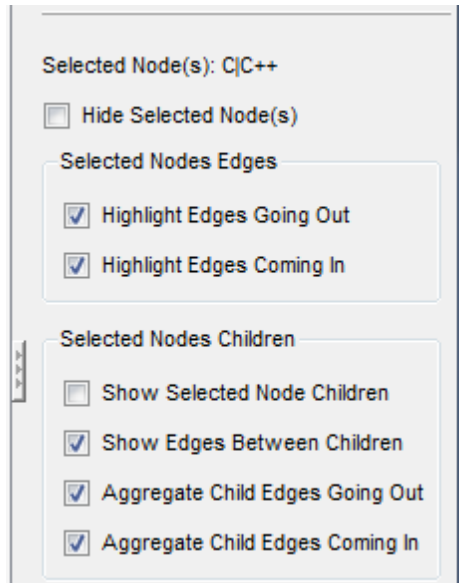
• **Clear All Highlightened Edges:** 仅对内部依赖视图有效，勾选此选项的情况下，所有高亮显示的节点和“连接关系”被删除。

• **Show All Hidden Nodes:** 使用此选项将所有被隐藏的节点恢复显示，对于被隐藏的子节点无效。

• **Restore Defaults:** 使用此选项将视图恢复到原始状态。

在依赖视图中使用鼠标拖曳可以选取多个节点，也可以通过按下 Ctrl 键通过鼠标点击选取。

Selected Node(s)区域提供对选取节点的操作：



• **Hide Selected Nodes:** 通过此选项将选中节点隐藏，在需要的情况下 Understand 2.5 对视图进行重新组织，通过 **Show All Hidden Nodes** 恢复显示。

• **Show/Highlight Edges Going Out:** 勾选此选项将选中节点以黄色高亮显示，从此节点指向其他节点的箭头以深色显示，箭头指向的节点以亮蓝高亮显示。内部依赖视图将“连接关系”高亮显示，其余视图则控制“连接关系”的隐藏和显示。



• **Show/Highlight Edges Going In:** 勾选此选项将选中节点以黄色高亮显示，从其他节点指向此节点的箭头以深色显示，指向此节点的其他节点以亮蓝高亮显示。内部依赖视图将“连接关系”高亮显示，其余视图则控制“连接关系”的隐藏和显示。



• **Show Selected Nodes Children:** 勾选此选项将选中节点的子节点显示，与双击节点动作效果相同。

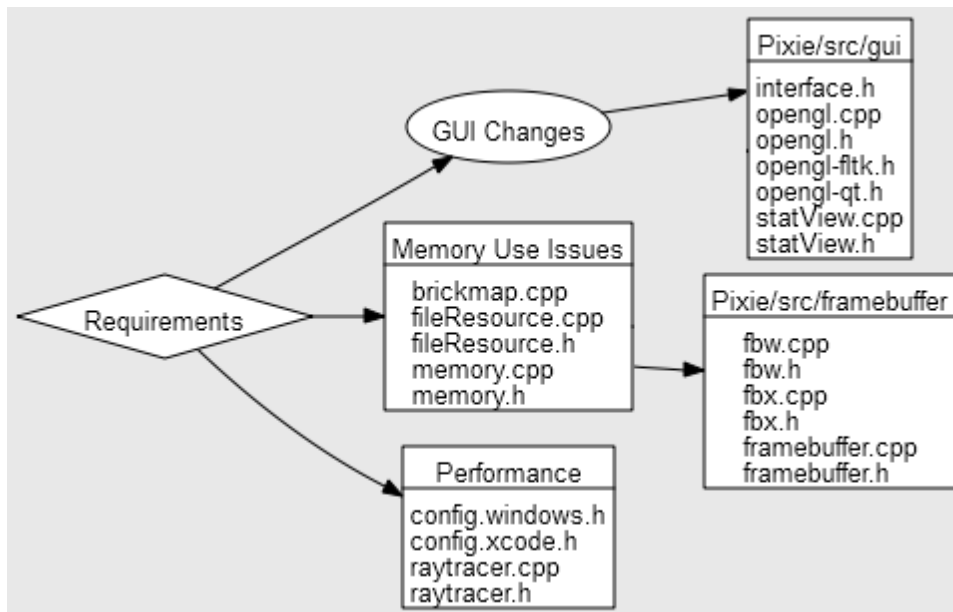
• **Show Edges Between Children:** 勾选此选项将选中子节点的相互关系以箭头标识，删除这些箭头会引发视图重新组织。

• **Aggregate Child Edges Going out:** 子节点显示的情况下勾选此选项，子节点引出的箭头被体现为节点引出，拥有相同目标的多个子节点关联的箭头被聚合成一个箭头。取消此选项恢复箭头与子节点的关联关系。

• **Aggregate Child Edges Going In:** 子节点显示的情况下勾选此选项，指向子节点的箭头被体

现为指向节点，来自相同节点指向本节点内多个子节点的箭头被聚合成一个箭头。取消此选项恢复箭头与子节点的关联关系。

**Graph Architecture** 视图没有提供定制面板，但是可以通过对依赖视图的右键菜单对显示进行修改。例如，在下面的层级结构视图，通过右键菜单将默认关闭的 **Include Entity Lists** 选项打开。



## 查看层级结构度量

Understand 2.5 支持生成层级结构或者其子集的度量数据，结果可以保存为文本格式或者以逗号分隔的电子数据表格。

通过以下步骤创建度量数据：

- 1 选中期望生成度量的节点；
- 2 在右键菜单中选择 **Metrics Summary**；
- 3 弹出层级结构度量数据窗口。例如，以下两个窗口显示了使用复杂度层级结构度量比较高复杂度和低复杂度的数据结果；
- 4 关闭窗口时，弹出对话框询问是否需要保存，选择 **Save**，可以将结果保存到文本文件；



Architecture Metrics Summary			e Metrics Summary		
1	Complexity::Low - Architecture Metrics Summary				
2					
3	CountLineCodeExe	:	248.000000		66.000000
4	AvgCyclomaticStrict	:	1.315068		15.000000
5	CountStmtExe	:	253.000000		59.000000
6	CountLine	:	714.000000		116.000000
7	CountPath	:	96.000000		1536.000000
8	CountLinePreprocessor	:	3.000000		0.000000
9	CountSemicolon	:	272.000000		57.000000
10	CountStmt	:	292.000000		70.000000
11	AvgCyclomaticModified	:	1.315068		14.000000
12	CountInput	:	100.000000		6.000000
13	CountLineComment	:	91.000000		19.000000
14	CountOutput	:	132.000000		9.000000
15	MaxEssentialKnots	:	0.000000		11.000000
16	AvgCyclomatic	:	1.315068		14.000000
17	CountLineCodeDecl	:	112.000000		12.000000
18	CountLineBlank	:	100.000000		12.000000
19	CountLineInactive	:	1.000000		0.000000
20	MaxNesting	:	1.000000		2.000000
21	CountStmtDecl	:	39.000000		11.000000
22	CountLineCode	:	523.000000		90.000000
23					

创建度量导出文件，需要执行以下操作：

- 1 选中期望生成度量的节点；
- 2 在右键菜单中选择 **Metrics Export**；
- 3 弹出窗口以逗号分隔显示度量数据，首行显示列标签，各个子节点的度量数据以独立行显示。

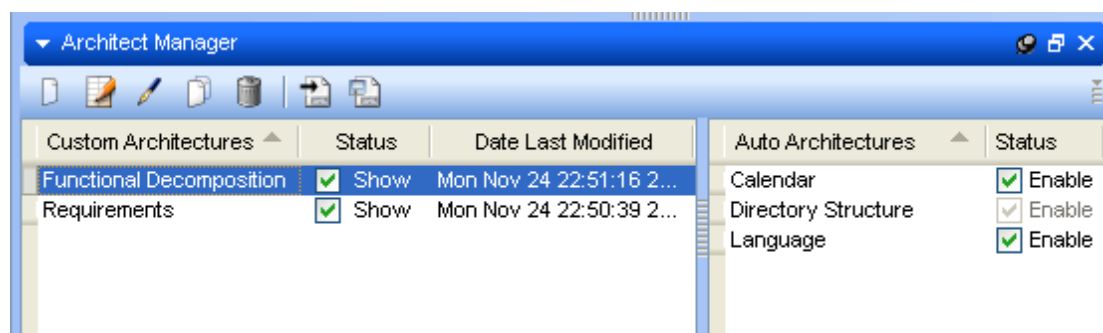
Complexity_MetricsExport.csv	
1	Name, CountLineCodeExe, AvgCyclomaticStrict, Cou
2	Complexity, 593, 2.45055, 599, 1535, 2016, 9, 597, 2..
3	Complexity::Low, 248, 1.31507, 253, 714, 96, 3, 272,
4	Complexity::Medium, 279, 6.58824, 287, 705, 384, 6, .
5	Complexity::High, 66, 15, 59, 116, 1536, 0, 57, 14, 70
6	

- 4 关闭窗口时，弹出对话框询问是否需要保存，选择 **Save**，将文件保存为 CSV 文件。

## 管理层级结构

通过菜单栏 **Project>Architectures>Manage Architectures** 打开层级结构管理窗口，左右两侧分别显示用户创建的层级结构和 Understand 2.5 提供的层级结构。

多选框可以控制层级结构显示与否，对于大型工程，隐藏层级结构可以显著提升效率，因此，对于极少或者从不使用层级结构的用户，可以选择隐藏层级结构的显示。





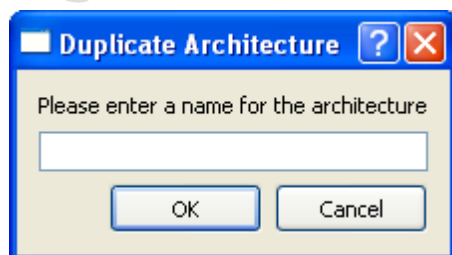
通过上图中的图标或者层级结构的右键菜单可以执行如下操作：


-  创建一个新的层级结构，参见[创建层级结构](#)。
-  编辑层级结构，参见[创建层级结构](#)。
-  层级结构重命名，参见[创建层级结构](#)。
-  复制层级结构，参见[创建层级结构](#)。
-  删除层级结构，参见[创建层级结构](#)。
-  从 XML 文件导入层级结构，参见[创建层级结构](#)。
-  导出层级结构到 XML 文件，参见[创建层级结构](#)。

## 创建层级结构


下面介绍了创建层级结构的两种方法：

- 在目录栏选择 **Project>Architectures>New Architecture**，或者点击层级结构管理器的  图标，可以新建一个层级结构，具体操作参见层级结构引导程序使用。
- 选中一个层级结构，点击层级结构管理器的  图标，可以拷贝一个现有的层级结构，然后对其进行修改。也可以在层级结构浏览器中右击一个层级结构节点，然后选择 **Duplicate Architecture**，对选中节点及其下层层级结构进行复制。

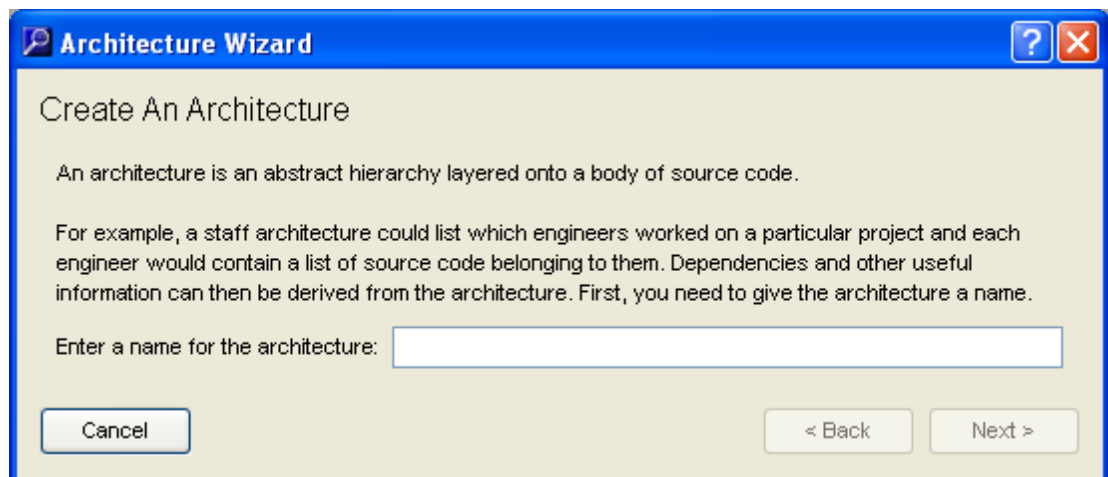


使用  图标可以对现有的层级结构进行重命名，或者对层级结构使用右键菜单选择 **Rename Architecture** 也可以实现同样操作。

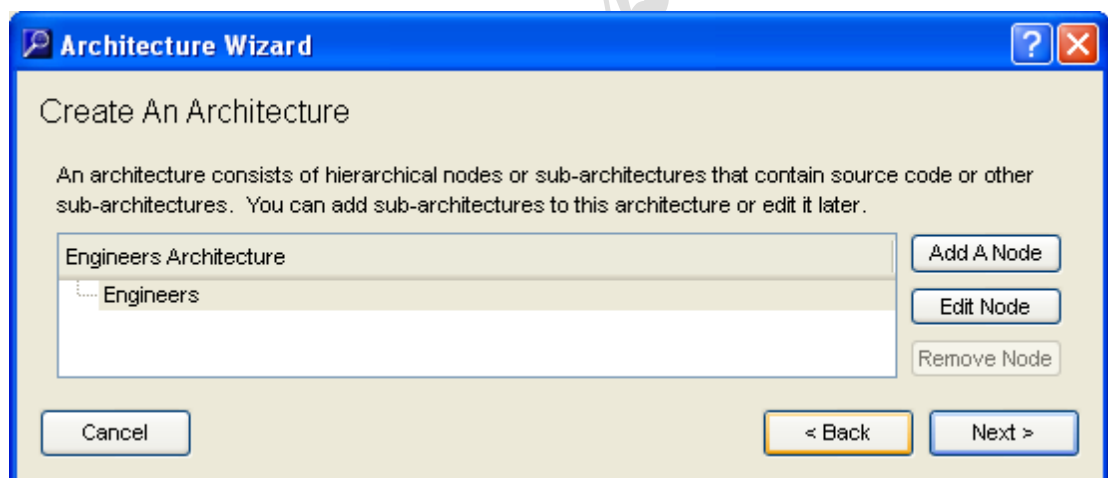
## 层级结构引导程序使用

通过 Project>Architectures>New Architecture，或者层级结构管理器中的  图标打开层级结构引导程序，首先显示层级结构名称设置页面。

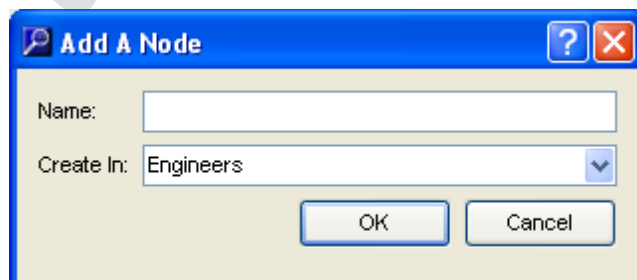
输入层级结构名称，可以尽量简短以便能够在层级结构树完整显示。



点击 Next 跳转到节点管理页面，添加或者编辑层级结构节点，之后选择的实体会被按照期望分配到各个节点。



点击 **Add a Node**，输入期望的节点名称，节点默认添加到引导程序选中的节点，也可以通过 **Create In** 对节点位置进行修改，然后点击 **OK**。





选中一个节点点击 **Edit Node** 可以对其进行修改，也可以通过 **Remove Node** 将其删除。

接下来的窗口通过动画展示如何通过层级结构构造器将实体分配到用户创建的节点。结束动画展示后，点击 **Finish**，会自动弹出层级结构构造器，用户创建的节点在右侧显示，添加实体

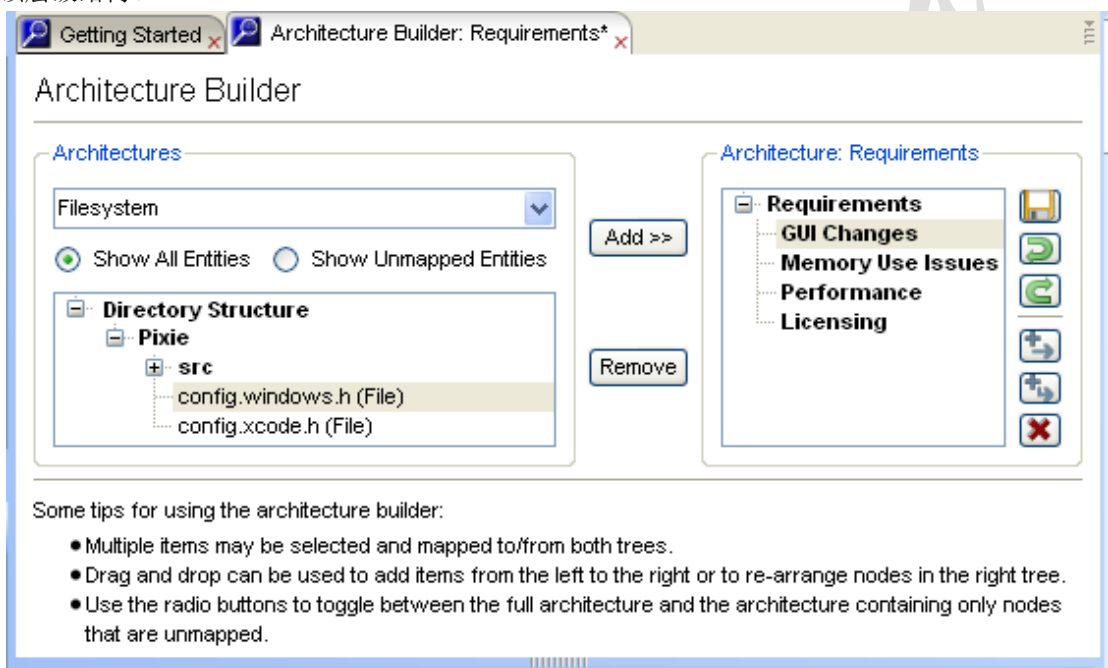
的方法参见[构建层级结构](#)。

## 构建层级结构






选中一个层级结构，点击层级结构管理窗口的  图标，对现有的定制层级结构进行编辑。层级结构的右键菜单中 **Edit Architecture** 也可以实现同样操作。两个操作均打开层级结构构造器。

Understand 2.5 提供的层级结构不允许修改，但是用户可以通过  对这些层级结构进行复制，对其副本进行编辑。

用户通过下面的对话框向层级结构添加节点，在右侧创建一个层级结构，从左侧将实体映射到该层级结构。



层级结构构造器中创建和编辑节点的方法如下：

- 双击层级结构构建器右侧任意节点的名称，可以为该节点重新命名（或者选择一个节点，然后敲回车）。
- 将一个或者多个节点拖动到期望的父节点，节点的所有子节点按照字母序排列。
- 点击  创建一个与所选节点同级的新节点，点击  创建一个所选节点的子节点。
- 点击  删除选中节点。
- 点击  撤销最近一次修改，点击  重做上一次撤销。

按照如下步骤可以在层级结构构建器将文件映射到节点：

- 1 在层级结构构建器左侧窗口，从包含各个文件名称的下拉列表选择一个已存的层次结构，默认为目录结构层次结构。
- 2 选择显示层次结构的所有条目或者未映射条目。例如，如果期望将所有条目映射到新建的层次结构，可以选择 **Show Unmapped Entries** 将未映射的文件全部显示出来。
- 3 在左侧的层次结构图中，选择一个或多个文件或者其中一个节点。


---

4 在右侧的层次结构图，选择期望包含左侧选中内容的对应节点。

5 点击 **Add** 或者将选中内容拖动到右侧窗口。

6 完成层次结构的编辑后，点击 **Save**。

使用 **Remove** 可以将层次结构图中的文件或者节点删除。层次结构图中的文件和节点同样支持右键菜单。

任意时刻点击按钮  将对层次结构图的修改保存，之后可以继续进行修改。存在未保存修改的情况下关闭层次结构构建器，Understand 2.5 会自动弹出是否需要保存的提示窗口。


## 使用 XML 管理层级结构

通过使用 XML 可以将一个层级结构在多个 Understand 2.5 数据库之间共享。除了共享功能，Understand 2.5 还支持对一个层次结构图中的节点通过 XML 导出/导入来快速创建层级结构。

### 导出 XML 格式的层级结构

通过如下步骤创建一个层级结构的 XML 文件：

1 选中期望导出的层级结构的最上层节点，将导出包含该层级结构所有层次结构图信息的 XML 文件。

2 点击层级结构管理器中的  按钮，或者右击对象节点选择 **XML Export**。

3 Understand 2.5 创建 XML 文件，包含 <arch>和<tag>来指示层级结构的节点。

4 关闭 XML 文件，自动弹出是否需要保存的提示窗口，默认保存文件名与节点名称一致，点击 **Save**。

### 导入 XML 的层级结构

通过如下步骤根据 XML 文件创建一个层级结构：

1 点击层级结构管理器中的  按钮。

2 在弹出的层级结构导入对话框，选择一个符合 Understand 2.5 标记定义的描述层级结构的 XML 文件，如 Understand 创建的一些 XML 文件，点击 **Open**。

3 XML 文件描述的层级结构被加入到当前工程层级结构列表。

---

## 报告

本章介绍如何创建和查看 **Understand** 报告，以及 **Understand 2.5** 报告支持的类型。

本章包含以下内容：

- 配置报告
- 生成报告
- 查看报告
- 报告分类概述
- 交叉引用报告
- 结构报告
- 质量报告
- 度量报告

Shiningstone provided

## 配置报告

Understand 2.5 支持生成多种关于代码各方面的报告，这些报告以 HTML 格式或者文本格式输出，用户可以根据需要自由选择。

对报告进行配置，选择 **Reports>Configure Reports** 打开工程配置对话框，选择 **Reports>Output** 分类。或者，也可以通过 **Reports>Options** 和 **Reports>Selected** 进行配置。

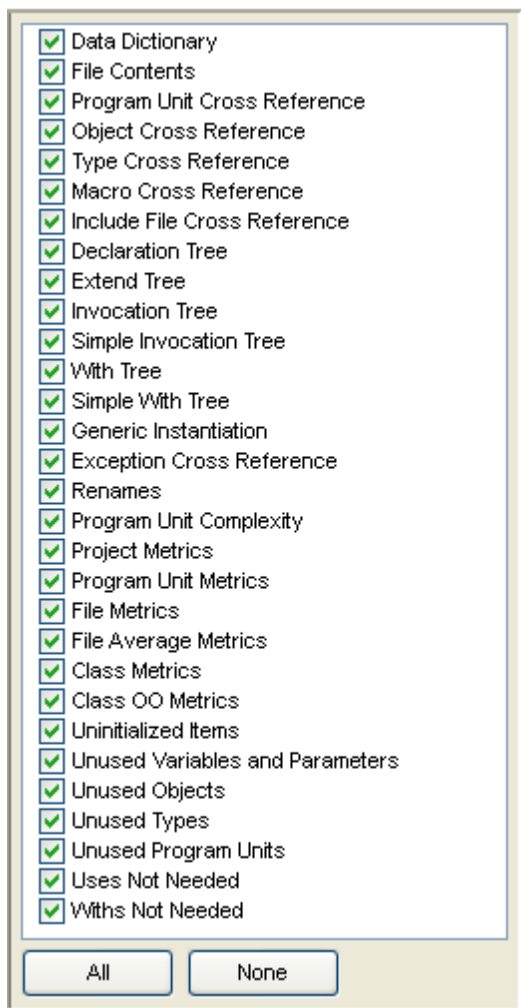
**Reports>Output** 分类的配置介绍参见 [Reports>Output](#)。大体来说，用户可以对选择如下输出：

- **HTML 报告**：报告首页为 index.html，用户也可以根据自己需要更改标题页面。每种类型的报告可以生成一个或者多个 HTML 文件，对于大型工程建议按照文件进行生成。选择 **Alphabetic** 将为报告生成的多个 HTML 文件按照实体首字母的字母序进行排列，选择 **Every n Entities** 控制报告生成的每个 HTML 文件包含固定数目 n 个实体。默认情况下，HTML 报告按照字母序生成。

- **文本报告**：生成一个单独的文本文件，其中包含所有选中的报告。同样也支持生成多个文本文件（选中 **Separate Files**），指定共用的文件名前缀，文件名指示生成报告内容。

**Reports>Options** 详细介绍参见 [Reports>Options](#)。

**Reportss>Selected** 提供了当前工程能够使用的所有报告类型供用户选择，下图列出了所有报告类型：



具体的可选报告类型依赖于工程使用的语言类型。各种报告类型的介绍参见报告分类介

---

绍。

## 报告颜色定制

HTML 报告使用了层级样式表 (CSS) 来控制关键词, 注释, 字符串, 数字等不同类型的内容的颜色和字体显示, 相关设置在第一次生成 HTML 报告时自动创建的 `sourcestyles.css` 中定义。

`Sourcestyles.css` 可以通过文本编辑器修改, CSS 支持的颜色或者字体可以随意使用, 例如:

```
span.comment{color:DarkSeaGreen;font-style:italic}
```

如果期望将修改的样式表应用到其他生成的 HTML 报告, 将该文件拷贝至其他 HTML 报告所在的路径即可。

## 生成报告

对报告格式和类型进行配置后, 选择菜单栏 **Reports>Generate Reports** 即可打开报告生成过程控制对话框。

对于 windows 操作系统, ASCII 文本报告遵循 DOC 文本文件格式定义, 对于 UNIX 操作系统, 则按照 UNIX 惯例 (行尾附加回车符) 生成文本类型的报告。

HTML 格式报告以 HTML3.0 格式生成, 支持 Netscape 2.0 及 Internet Explore 3.0 及以上版本。

查看报告参见[查看报告](#)。

对于大型工程, 生成报告可能会花费较长时间, 在生成过程中点击 **Cancel** 可以直接退出, 这个操作将导致报告不完整。

### 注意:

生成报告之前临时关闭防病毒程序能够加速操作, 请注意完成后恢复防病毒程序。

### 注意:

通过 “und” 命令程序也可以生成 HTML, 文本以及工程度量报告, 详细信息参见第 13 章。

## 查看报告

点击 **Reports>View Reports**, 然后选择 **HTML** 或者 **Text** 即可查看生成的报告。

报告名称根据生成报告的类型和保存格式确定。

- 对于文本格式, 可以将所有选中报告类型包含在一个文件, 或者针对每种报告类型生成一个报告文件, 单个文件的报告以 `<project_name>.txt` 命名, 多个文件的报告根据生成报告类型进行命名。

- 对于 HTML 格式, 可以针对每种报告类型生成一个 HTML 报告, 也可以根据字母或固定实体数目将报告文件分得更小。同时还会生成一个索引文件, 包含对其他所有报告的链接, 其默认名称为 `index.html`。

对于 HTML 报告, 包含一个索引文件, 以字母序排列列出所有报告中的实体, 每个实体链接到对应实体的数据字典报告。这个索引文件被命名为 `entity_index.html`, 可以通过主 HTML 页面



的 index 链接访问。  
下图列出了一个实体索引的例子：



## 报告分类概述

- Understand 2.5 支持多种类型的报告：
- **Cross-Reference** 报告包含信息浏览器中的相近信息，并字母序将所有实体进行排列，参见交叉引用报告。
  - **Structure** 报告显示分析项目的结构，参见结构报告。
  - **Quality** 报告显示需要进行检查的代码区域，参见质量报告。
  - **Metrics** 报告显示代码和注释行数等基本度量信息，参见度量报告。

报告类型	报告名称和页码
交叉引用	数据字典报告，参见
交叉引用	文件内容报告，参见
交叉引用	程序单元交叉引用报告，参见
交叉引用	对象交叉引用报告，参见
交叉引用	类型交叉引用报告，参见
交叉引用	宏交叉引用报告，参见
交叉引用	头文件交叉引用报告，参见
交叉引用	异常交叉引用报告，参见
结构	声明树，参见
结构	类扩展树，参见
结构	调用树报告，参见
结构	简单调用树报告，参见
结构	导入报告，参见
结构	支持树报告，参见
结构	简单支持树报告，参见

结构	类实例报告，参见
结构	重命名报告，参见
质量	程序单元复杂度报告，参见
质量	未初始化条目报告，参见
质量	未使用变量和参数报告，参见
质量	未使用对象报告，参见
质量	未使用类型报告，参见
质量	未使用程序单元报告，参见
质量	冗余支持报告，参见
质量	隐含声明对象报告，参见
质量	FORTTRAN 扩展使用报告
度量	工程度量报告
度量	程序单元度量报告
度量	文件度量报告
度量	文件平均度量报告
度量	类度量报告
度量	类 OO 度量报告

## PERL/C API

Understand 2.5 多年来致力于支持更加贴近用户的报告类型，当然还是不能满足多样化的需求。为了让用户能够自由地对报告进行定制，Understand 提供了访问数据库的 PERL 和 C 接口。

通过 **Help>PERL API** 可以详细了解 PERL 接口，Understand 网站的论坛和博客也提供了这方面的支持。

Reports>Project Interactive Reports 和 Graphs>Project Graphs 显示了一些用户自定义的插件，通过 PERL API 可以使用这些插件。[创建插件的详细信息可以联系 support@scitools.com](#)，Understand 网站的论坛和博客也提供了这方面的支持。

## 交叉引用报告

交叉引用报告能够显示与信息浏览器中引用区域的相近信息，更为便捷的是交叉引用报告将所有实体以字母序排列显示在一起。下表列出各种交叉引用报告的位置。

[数据字典报告](#)

[程序单元交叉引用报告](#)

[文件内容报告](#)

[对象交叉引用报告](#)

[类型交叉引用报告](#)

[类和接口交叉引用报告](#)

[宏交叉引用报告](#)

[头文件交叉引用报告](#)

## 数据字典报告

数据字典报告按照字母序列出所有实体，包含实体名称，实体类型（如宏，类型，变量，函数，头文件包含，文件或者过程），数据字典条目提供到代码对应位置的链接。

<a href="#">Non-Alpha</a>	<a href="#">A</a>	<a href="#">B</a>	<a href="#">C</a>	<a href="#">D</a>	<a href="#">E</a>	<a href="#">F</a>	<a href="#">G</a>	<a href="#">H</a>	<a href="#">I</a>	<a href="#">J</a>	<a href="#">K</a>	<a href="#">L</a>	<a href="#">M</a>	<a href="#">N</a>	<a href="#">O</a>	<a href="#">P</a>	<a href="#">Q</a>	<a href="#">R</a>	<a href="#">S</a>	<a href="#">T</a>
---------------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------	-------------------

```
echo      (Procedure)\[xref\]  
           \[lexlib.pas, 284\]  
  
emp       (Variable)\[xref\]  
           \[dept\_empl.pas, 150\]  
  
employee  (Sql Table)  
           \[dept\_empl.pas, 9\]  
  
employee.age  (Sql Column)  
              \[dept\_empl.pas, 11\]
```

## 程序单元交叉引用报告

---

文件内容报告

对象交叉引用报告

类型交叉引用报告

类和接口交叉引用报告

宏交叉引用报告

头文件交叉引用报告

异常交叉引用报告

Shiningstone provided