

Qt6 C++开发指南学习笔记

第1章 认识Qt

1.1 Qt简介

- 1.1.1 Qt的基本情况
- 1.1.2 Qt的跨平台开发能力
- 1.1.3 Qt的许可类型和安装包
- 1.1.4 Qt支持的开发语言
- 1.1.5 Qt6新特性

1.2 Qt6安装

- 1.2.1 Qt6如何安装?
- 1.2.2 可能遇到的错误

1.3 Qt Creator初识

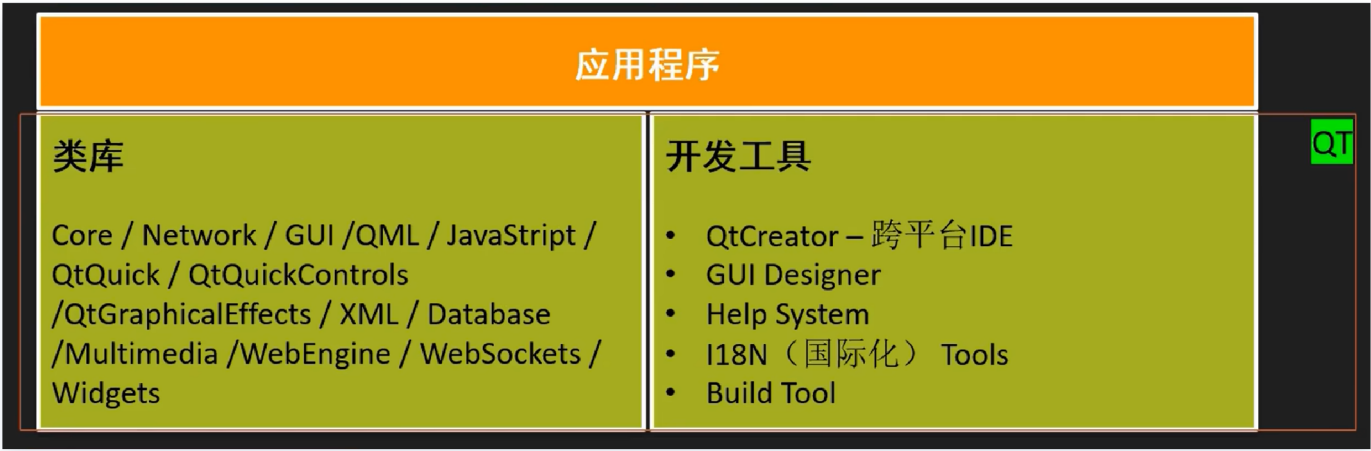
- 1.3.1 Qt Creator简介
- 1.3.2 Qt Creator 欢迎界面
- 1.3.3 Qt Creator 新建项目
- 1.3.4 Qt Creator 编辑 UI
- 1.3.5 Qt Creator 生成和运行程序

第1章 认识Qt

1.1 Qt简介

1.1.1 Qt的基本情况

- Qt是一个跨平台的应用程序开发框架。也是最主流的C++开发框架
- Qt于1995年5月首次公开发布。
- Qt官网: www.qt.io
- Qt具有其他编程语言的扩展, 但Qt本身是用 C++ 开发的



1.1.2 Qt的跨平台开发能力

只要熟悉一种平台的开发，就能很快适应其他平台的开发。并且Qt源代码编译后生成 目标平台的原生二进制代码，不像Java那样需要虚拟机， 运行效率更高。

目标设备	目标平台（运行平台）	主机平台（开发平台）
计算机	桌面Linux macOS Windows	桌面Linux macOS Windows
移动设备	Android iOS 手机Windows	桌面Linux、 macOS、 Windows macOS Windows
嵌入式设备	嵌入式Linux QNX VxWorks 嵌入式Windows	桌面Linux 桌面Linux、 Windows 桌面Linux、 Windows Windows
单片机	FreeRTOS或无操作系统	Windows、 桌面Linux

注： 目标平台和主机平台每一行都一一对应

1.1.3 Qt的许可类型和安装包

- Qt的许可类型：
 - 商业许可： 按年付费， 模块更多， 开发者可以不公开自己的源码。
 - 开源许可：
 - GPLv2/GPLv3许可： 使用了GLP许可的Qt代码允许销售但必须开源。GPLv3还要求公开相关硬件信息。

- LGPLv3许可：与GPL类似，但更宽松。若只是链接或调用GPL许可协议的Qt代码，可以不开源。
- Qt安装包：根据目标设备不同，提供了不同的安装包。

1.1.4 Qt支持的开发语言

1. C++和QML:

- Qt对标准C++语言进行了扩展，引入了信号与槽等机制。（本教程内容）
- QML(Qt Meta Language)是一个用来描述应用程序界面的 **声明式脚本语言**。
- C++和QML可以混合使用。

2. Python

- Qt C++可以被转换为Python绑定，使用Python调用Qt类库进行GUI程序开发。
- 常用的Qt类库Python绑定有：PyQt(GPLv3许可)和PySide(LGPLv3许可)

1.1.5 Qt6新特性

- 2020年12月正式发布了Qt6.0,引入的主要新特性包括：
 - 支持C++17标准。
 - Qt核心库的改动：新的属性和绑定系统；全面支持Unicode字符串；修改了QList的实现方式，将QVector类和QList类统一为QList类；QMetaType和QVariant,几乎被重写。
 - 新的图形架构：Qt5中3D渲染依赖OpenGL.Qt6中引入了新的技术RHI(rendering hardware interface), 它是一个抽象层，使得Qt可以使用平台本地化的3D图形API (Linux: Vulkan, Windows: Direct 3D, macOS: Metal)
 - CMake构建系统：Qt6仍然支持gmake但建议使用CMake
 - 其他（多媒体、网络、Qt Quick3D等模块的改进

1.2 Qt6安装

1.2.1 Qt6如何安装?

- 在线安装包下载地址：https://download.qt.io/official_releases/online_installers/

Name	Last modified	Size	Metadata
📁 snapshots/	22-Nov-2022 10:06	-	
📁 online/	19-Nov-2020 14:24	-	
📁 official_releases/	01-Dec-2022 09:12	-	
📁 new_archive/	29-Jan-2021 15:13	-	
📁 ministro/	20-Feb-2017 10:32	-	
📁 linguist_releases/	26-Mar-2019 07:49	-	
📁 learning/	24-Feb-2021 15:09	-	
📁 development_releases/	06-May-2021 07:48	-	
📁 community_releases/	23-Feb-2017 07:29	-	
📁 archive/	01-Feb-2023 14:33	-	
📄 timestamp.txt	31-Mar-2023 04:00	11	Details

CSDN @qq_53873381

Name	Last modified	Size	Metadata
📁 Parent Directory		-	
📁 vsaddin/	24-Feb-2023 11:51	-	
📁 qtdesignstudio/	13-Jul-2022 20:21	-	
📁 qtcreator/	28-Mar-2023 11:22	-	
📁 qtchooser/	08-Oct-2018 07:53	-	
📁 qt3dstudio/	28-Oct-2020 14:22	-	
📁 qt/	29-Sep-2022 07:41	-	
📁 qt-installer-framework/	13-Mar-2023 09:55	-	
📁 qbs/	23-Feb-2023 11:39	-	
📁 pyside/	30-Nov-2015 13:39	-	
📁 online_installers/	13-Mar-2023 11:43	-	
📁 jom/	12-Dec-2018 15:13	-	
📁 gdb/	17-Nov-2014 13:42	-	
📁 additional_libraries/	03-Mar-2021 10:18	-	
📁 QtForPython/	12-Apr-2022 15:00	-	
📄 timestamp.txt	31-Mar-2023 04:00	11	Details

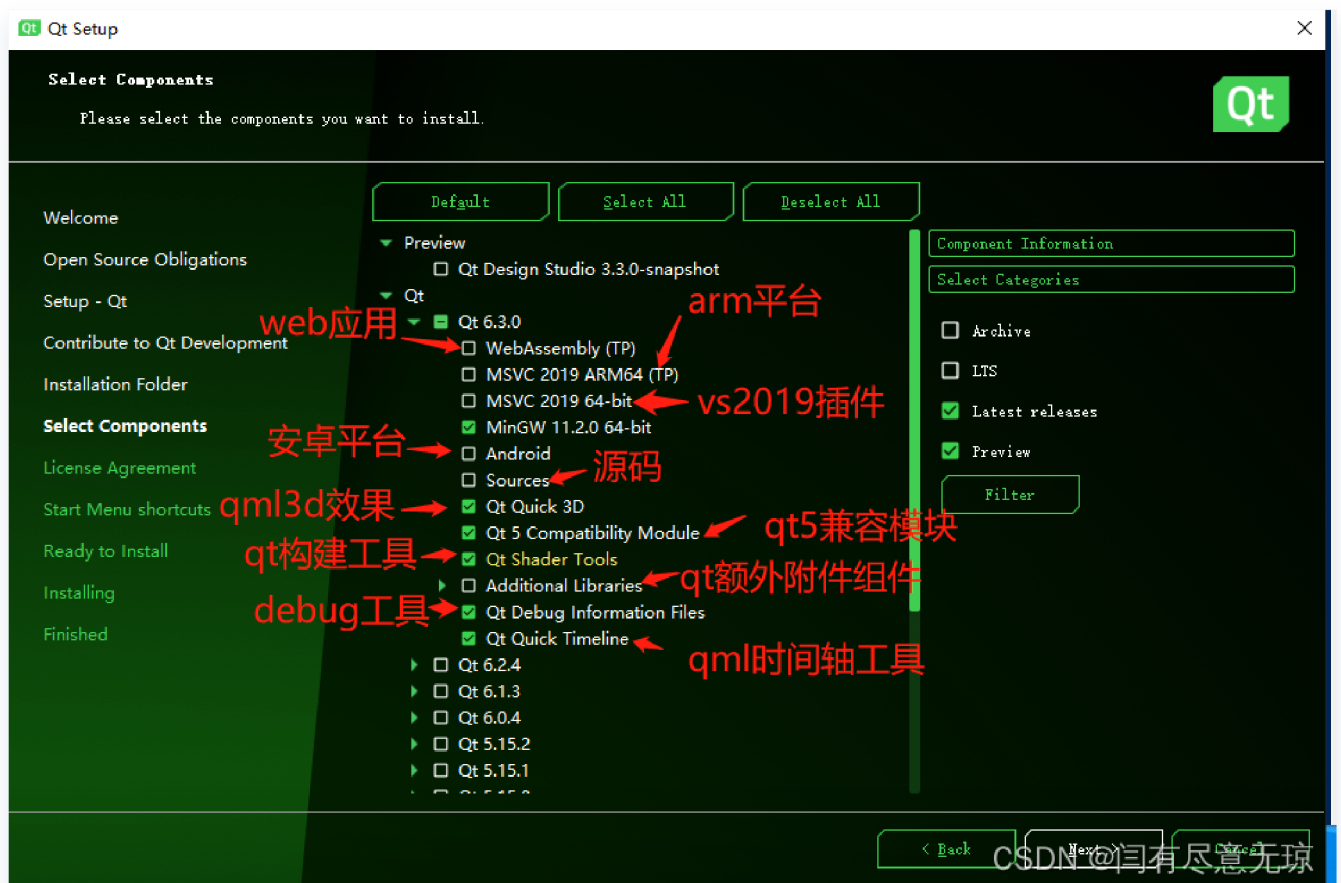
CSDN @qq_53873381

- 根据操作系统选择对应的下载文件

Name	Last modified	Size	Metadata
↑ Parent Directory		-	
qt-unified-windows-x64-online.exe	13-Mar-2023 10:04	41M	Details
qt-unified-mac-x64-online.dmg	13-Mar-2023 10:04	18M	Details
qt-unified-linux-x64-online.run	13-Mar-2023 10:04	55M	Details

CSDN @qq_53873381

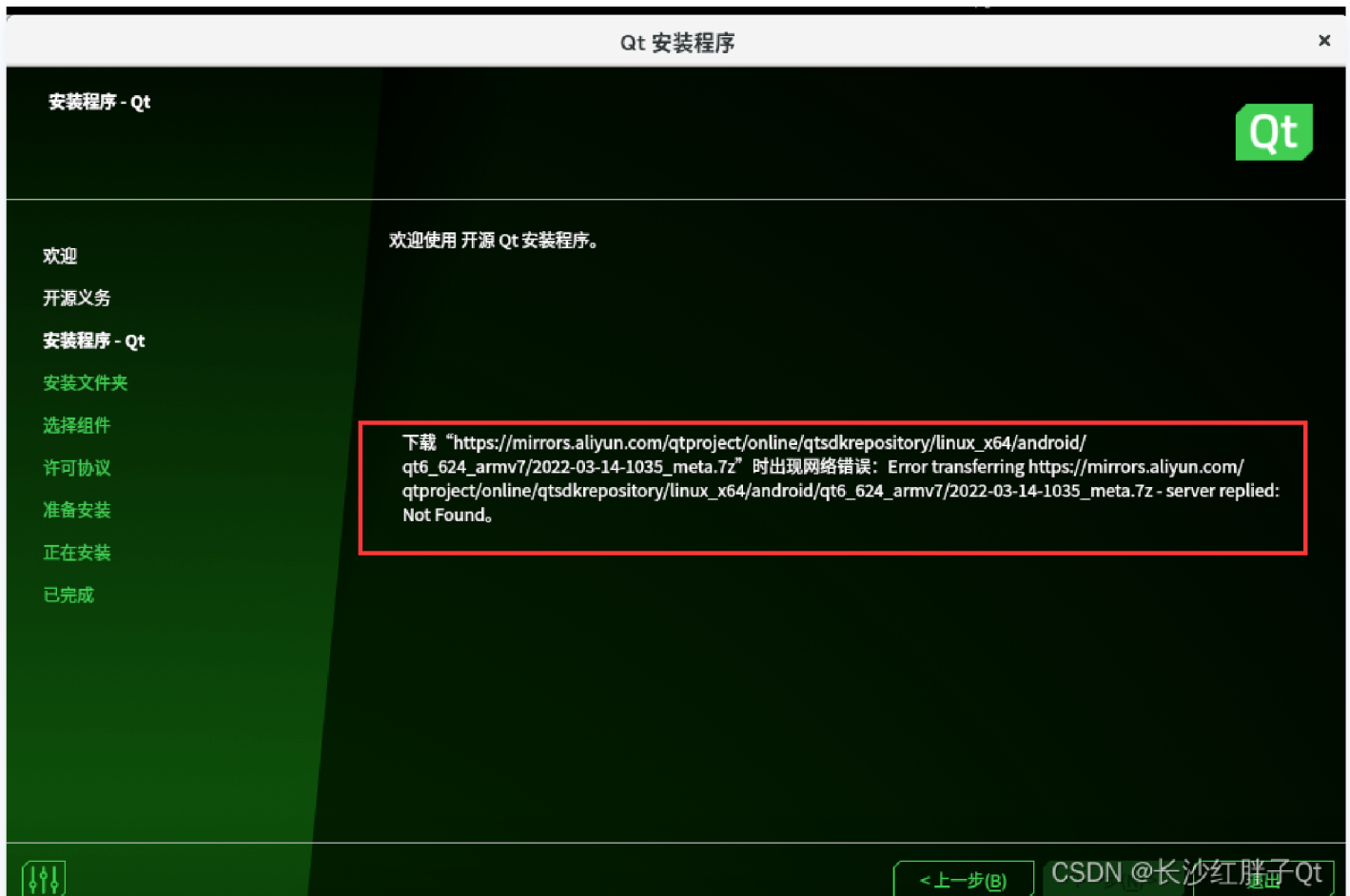
- 只需要安装需要使用的组件，后期可以添加删除。



- 运行Qt Creator,打开一个自带例子进行测试。

1.2.2 可能遇到的错误

qt国内下载安装过程中可能会遇到镜像获取不到的错误,如下图所示:

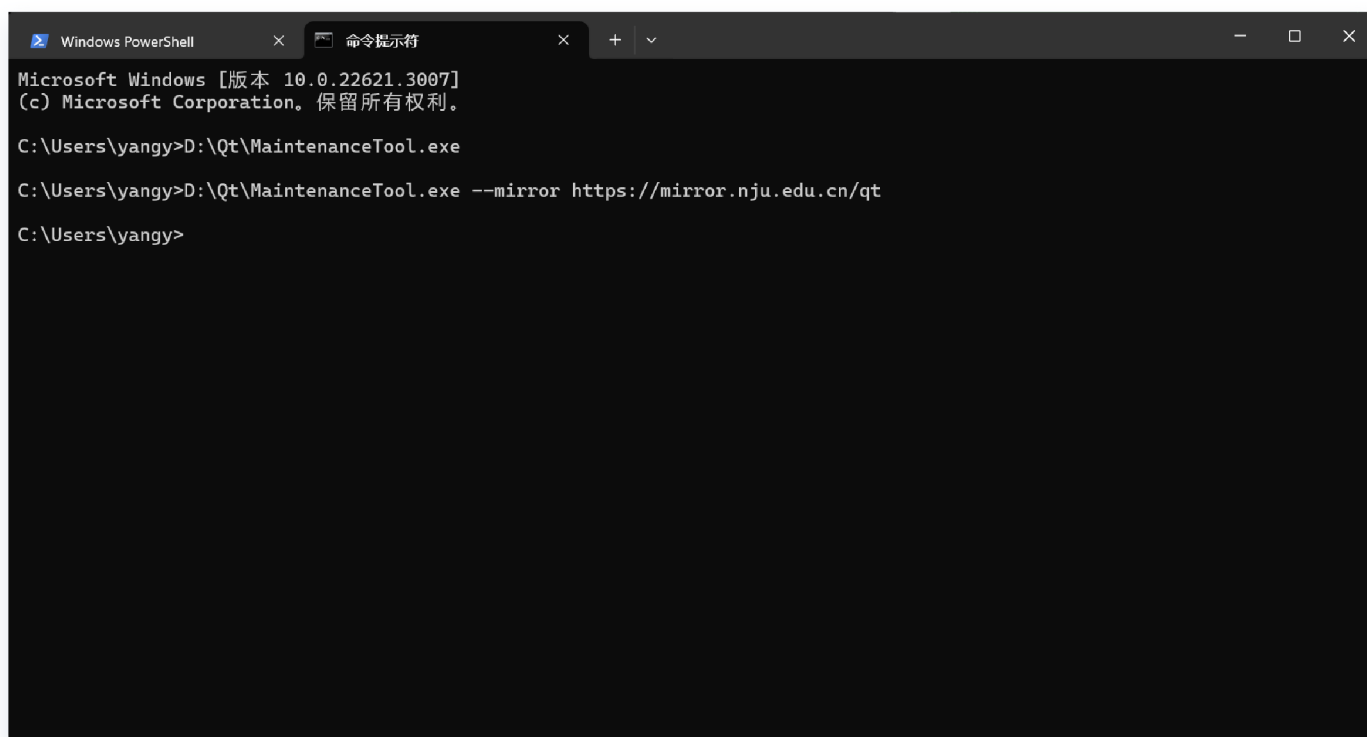


解决方法:更换国内的镜像,下面以Windows操作系统为例

- 在 [在线安装包](#) 的文件夹下打开 [cmd命令行](#)
- 手动输入安装包的文件名,也可直接将安装包拖入到cmd命令行中,并在后面添加参数 `--mirror` [国内镜像地址](#) ,例如

```
1 | MaintenanceTool.exe --mirror 国内镜像地址
```

这样, 安装程序就会尝试从国内镜像地址获取所需文件, 解决了因无法访问外网镜像而导致的下载错误。



```
Windows PowerShell
命令提示符

Microsoft Windows [版本 10.0.22621.3007]
(c) Microsoft Corporation。保留所有权利。

C:\Users\yangy>D:\Qt\MaintenanceTool.exe

C:\Users\yangy>D:\Qt\MaintenanceTool.exe --mirror https://mirror.nju.edu.cn/qt

C:\Users\yangy>
```

- 常用的国内镜像
 - 南京大学: <https://mirror.nju.edu.cn/qt>
 - 腾讯: <https://mirrors.cloud.tencent.com/qt/>
 - 中国科学技术大学: <http://mirrors.ustc.edu.cn/qtproject/>
 - 清华大学: <https://mirrors.tuna.tsinghua.edu.cn/qt/>
 - 北京理工大学: <http://mirror.bit.edu.cn/qtproject/>
 - 中国互联网络信息中心: <https://mirrors.cnnic.cn/qt/>

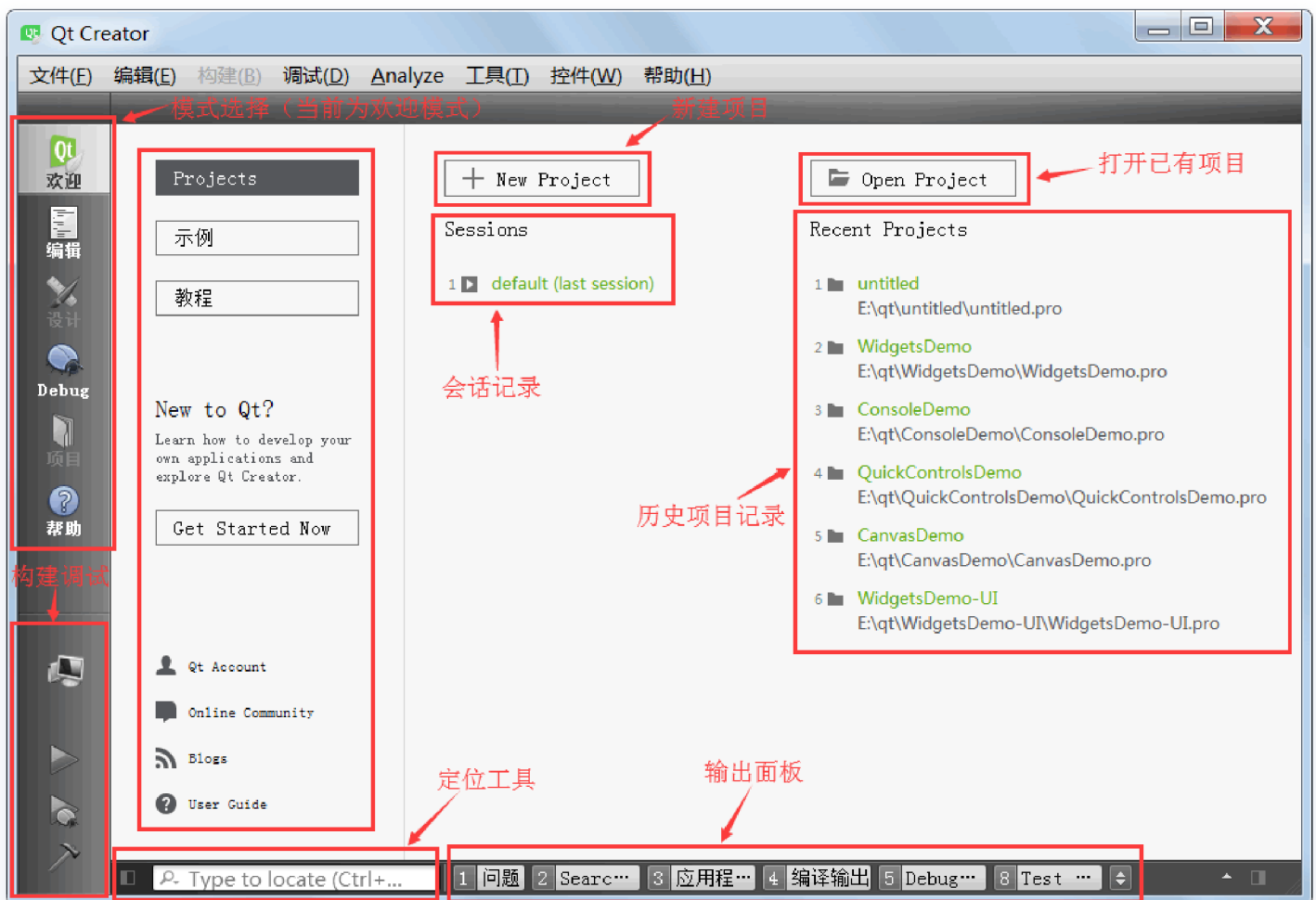
1.3 Qt Creator初识

1.3.1 Qt Creator简介

- Qt Creator为应用程序开发人员提供了一个完整的 跨平台、集成开发环境(IDE) 以便为 桌面、 嵌入式 和 移动设备平台(如Android和iOS) 创建应用程序。
- Qt Creator提供了在整个 应用程序开发生命周期 所需的工具, 从创建项目到将应用程序部署到目标平台

1.3.2 Qt Creator 欢迎界面

从开始菜单或者快捷方式打开 QtCreator 集成开发环境, 启动之后看到类似下面的界面:



QtCreator 最左边是一排功能按钮。上半部分按钮是 QtCreator 工作模式选择，共有七种工作模式，分别是欢迎、编辑（编写代码）、设计（GUI可视化编辑）、Debug（调试程序）、项目（项目参数配置）、分析（程序执行效率分析）、帮助。下面四个按钮是构建调试区，由上到下依次是 Qt 套件选择、运行、调试运行和构建。

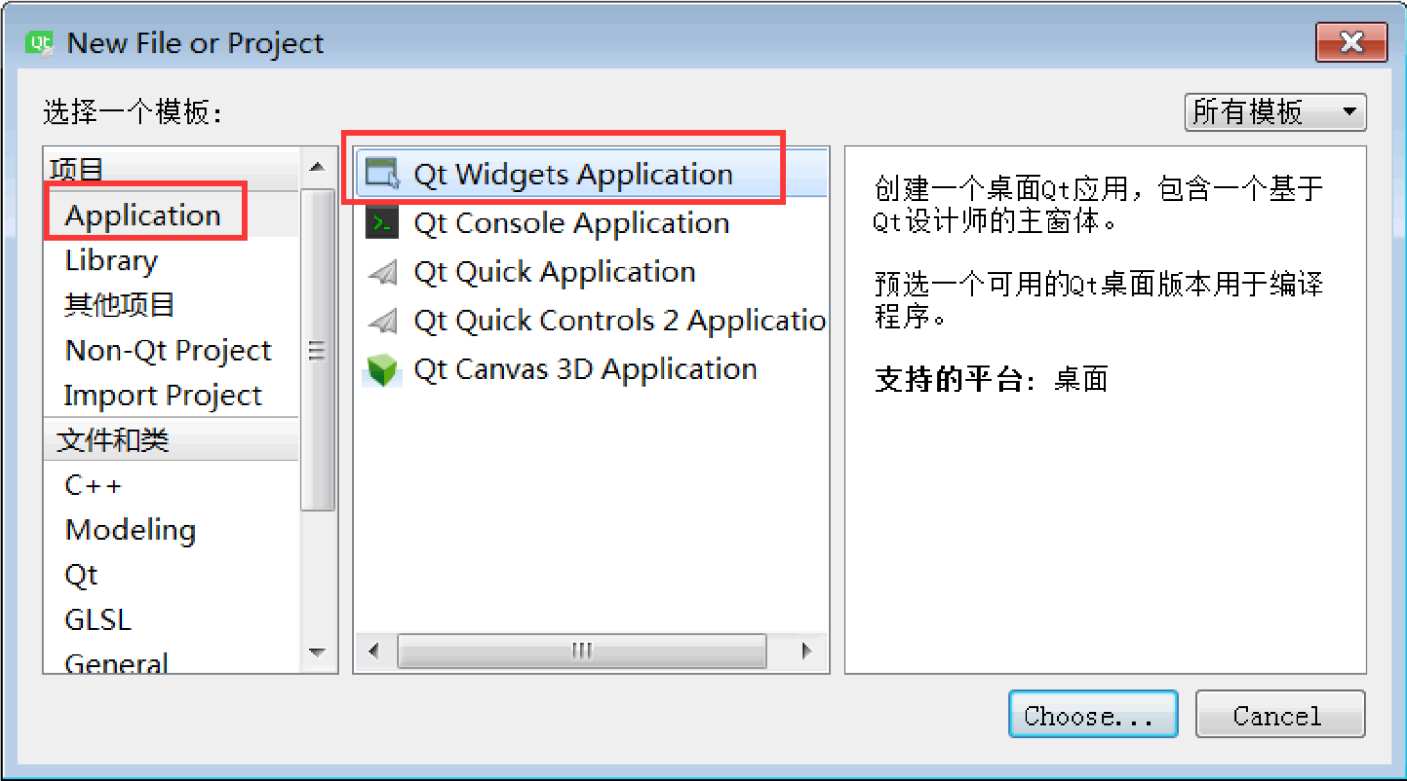
左边的设计按钮、项目按钮和构建调试区只有在打开或新建了项目之后才会变得可用。

QtCreator 下方的是定位工具和输出面板，在编写项目代码和运行、调试程序时会使用到。输出面板包括七个，分别是问题（项目构建时的问题）、Search Results（搜索项目文件内容）、应用程序输出（运行和调试信息显示）、编译输出（编译、链接命令及其输出信息）、QML/JS Console（QML 命令窗口）、概要信息（项目信息摘要）、Version Control（版本控制系统）。

QtCreator 中间的区域是所选择的工作模式界面，默认是欢迎模式。欢迎模式有三个子功能，第一个 Project 是项目显示，包括之前的会话和项目记录。项目记录比较好理解，而会话涵盖内容比较广，一个会话可以是多个项目的列表，并含有它们的配置以及上次编辑位置记录、调试断点等等。会话记录的上方是新建项目的快捷按钮，项目记录的上方是打开项目的快捷按钮。欢迎模式另外两个子功能是浏览 Qt 库自带的示例和教程，感兴趣的读者可以自行打开看看。

1.3.3 Qt Creator 新建项目

接下来我们新建一个 HelloWorld 项目。打开 Qt Creator 文件菜单，点击“新建文件或项目”（快捷键 Ctrl+N），或者直接在欢迎模式点击快捷按钮“New Project”，都可以打开如下所示的新建项目对话框：



新建项目对话框里有五类项目模板：

项目模板	说明
Application	Qt 应用程序，包括普通窗体程序和 QtQuick 程序。
Library	可以创建动态库、静态库以及 QtQuick 扩展插件、QtCreator 自身插件。
其他项目	可以创建单元测试项目、Qt4 设计师自定义控件、子目录项目等。
Non-Qt Project	非 Qt 项目。可以创建纯 C 或 纯 C++ 项目。
Import Project	导入项目。从版本控制系统管理的软件项目导入旧的项目。

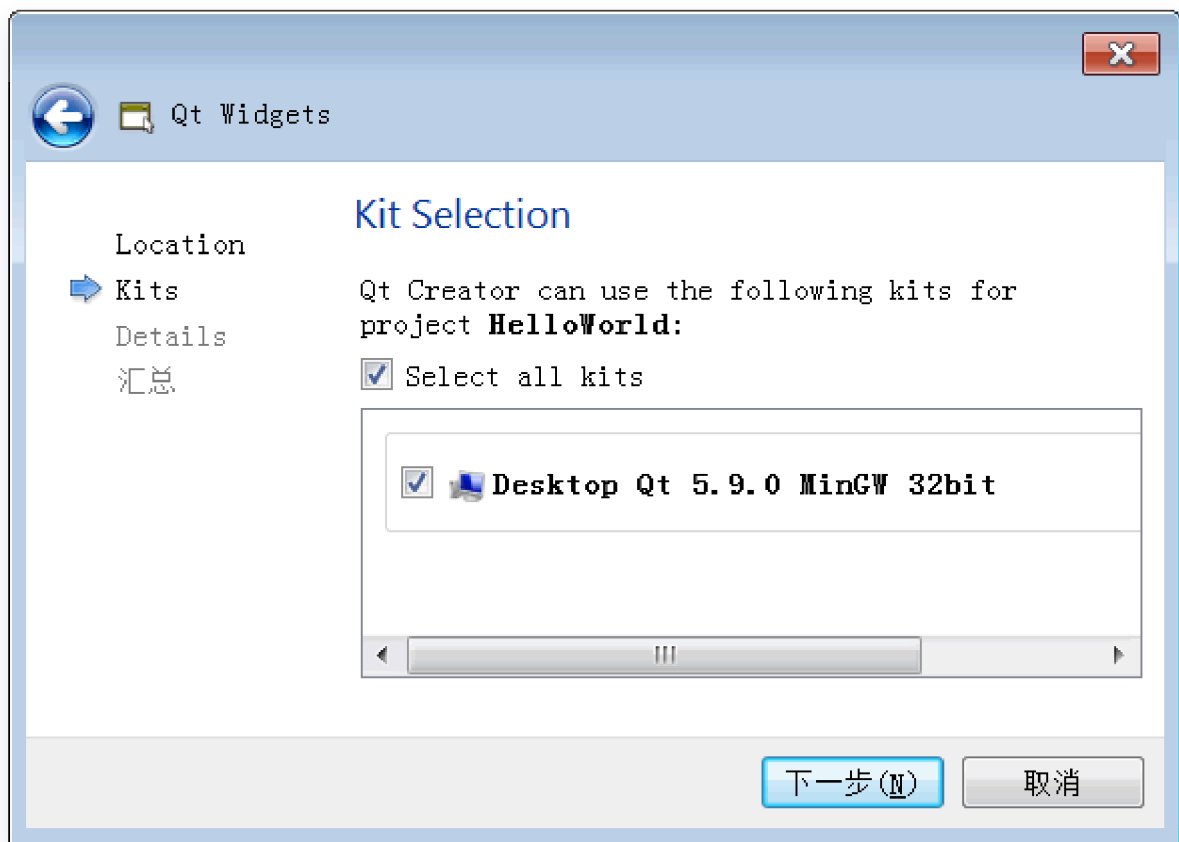
本教程常用的只有第一类 Application，选择它，在右侧会看到 Qt 应用程序的五个子模板：

- Qt Widgets Application：普通窗体模板，传统基于部件的窗体界面程序。
- Qt Console Application：Qt 控制台应用程序。因为 Qt 主要用于图形界面设计，这个控制台项目模板基本不用的。
- Qt Quick Application、Qt Quick Controls 2 Application 和 Qt Canvas 3D Application 工程将会使用 QML 代码，Quick Control 主要比 Quick 多一些控件，Canvas 3D 用来创建3D动画。

本教程使用第一个子模板 Qt Widgets Application，开发普通的 Qt 窗体应用程序。选择该项目模板，点击对话框下方的“Choose...”按钮，进入 Qt Widgets Application 项目新建的向导界面：

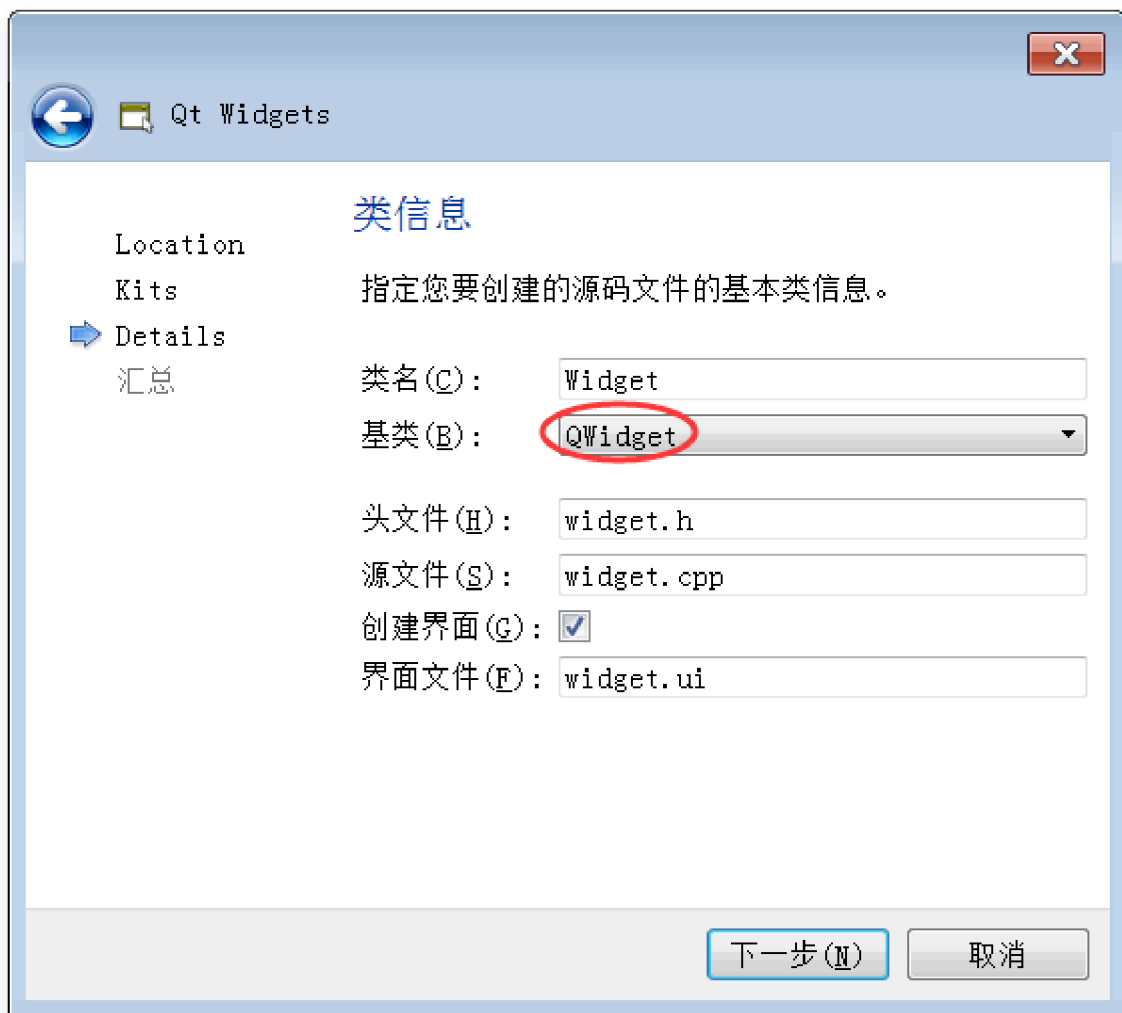


将项目名称设置为 HelloWorld，创建路径设置为 D:\QtDemo，点击“下一步”，进入“Kit Selection”界面：



这一步是为 HelloWorld 设置 Qt 套件（Qt Kits），默认只有第一个“Desktop Qt 5.9.0 MinGW 32bit”，如果安装配置了多个 Qt 套件，就可以都选上。

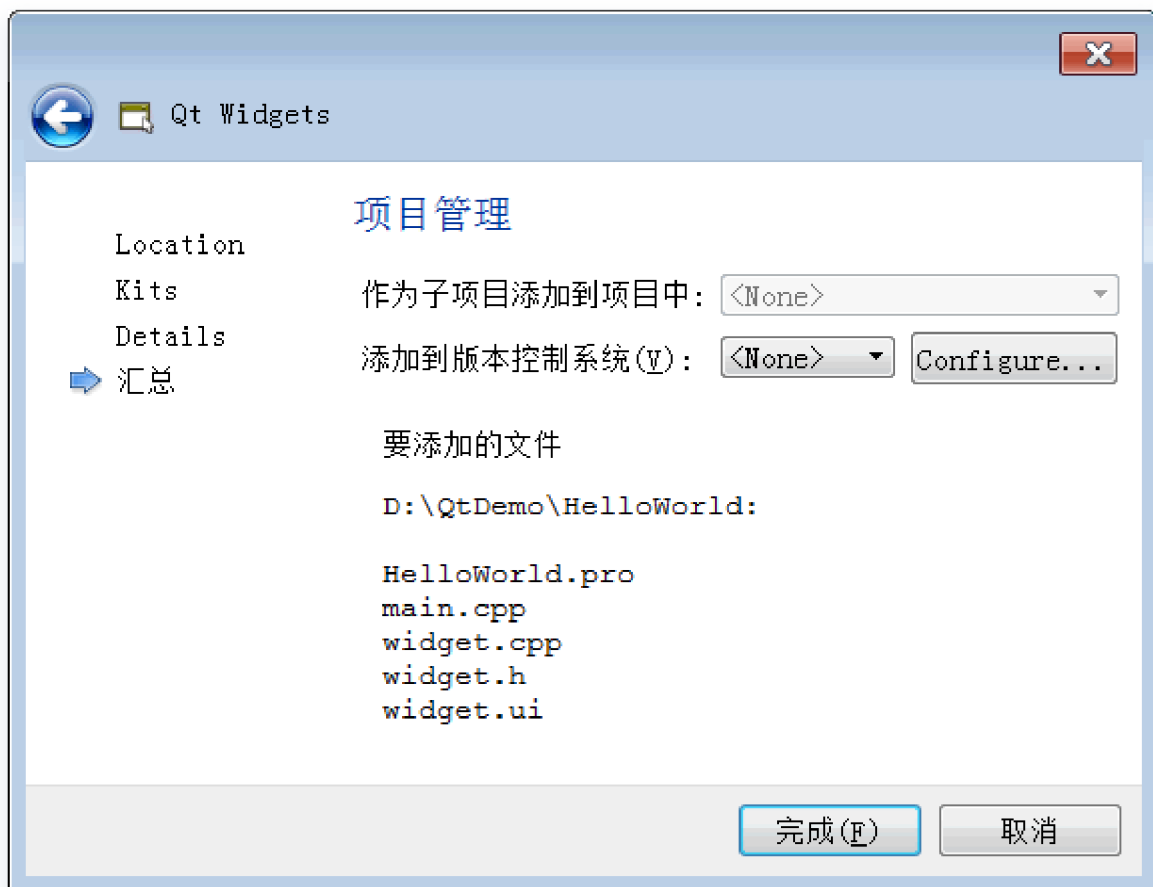
Qt 套件是指 Qt 程序从编译链接到运行环境的全部工具和 Qt 类库的集合，对于 MinGW 版本 Qt 程序生成和调试，至少需要 MinGW 中的编译器 g++（自动调用链接器）、g++ 配套的基础库、调试器 gdb 还有使用 MinGW 环境编译而成的 Qt 类库自身。默认情况下，在上面 Kit Selection 里选中全部套件，然后点击“下一步”，进入“类信息”设置界面：



类信息设置界面，最关键的是基类的选择，目前是三种基类：

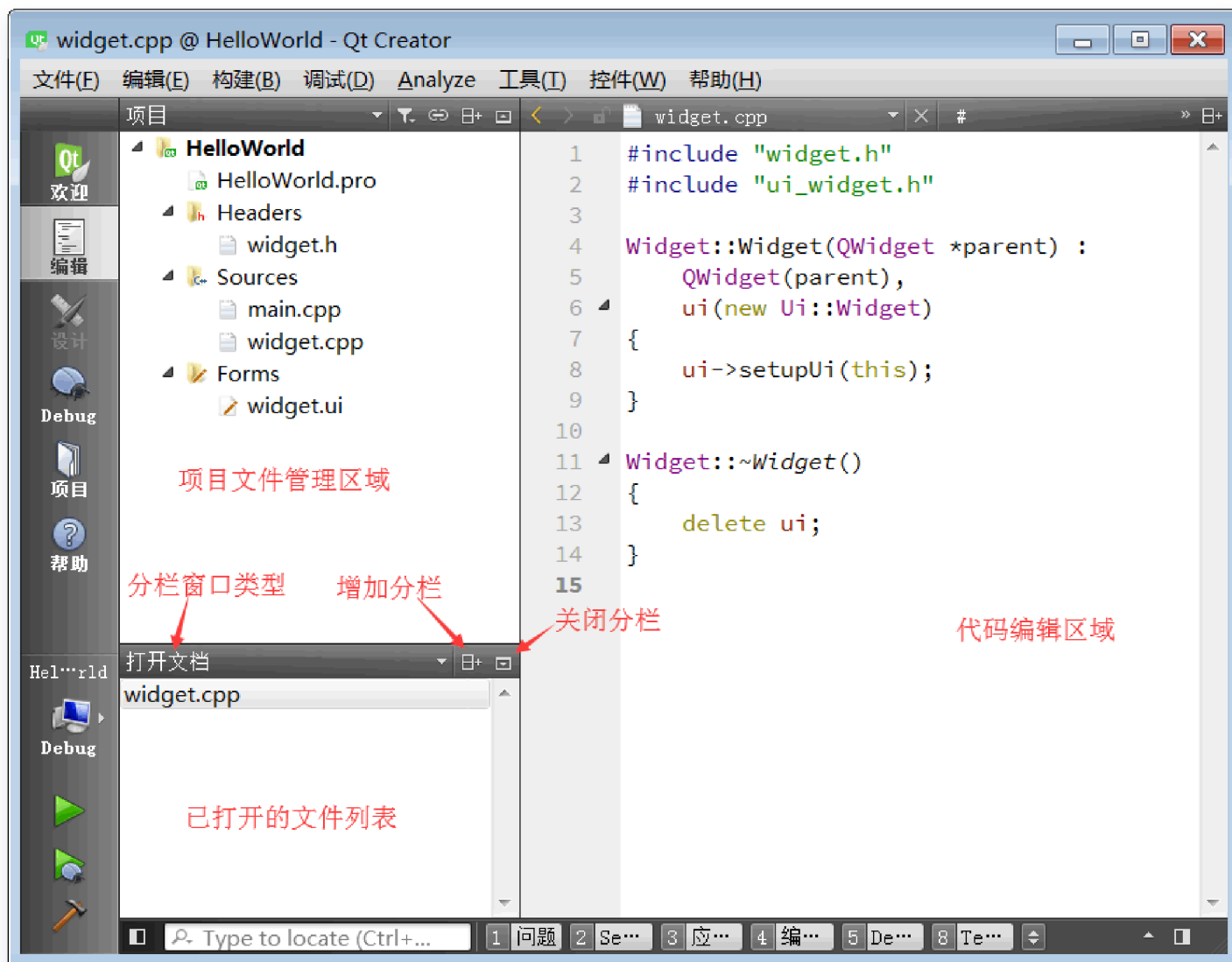
基类	说明
QMainWindow	基于主窗口类的程序，一般用于较为复杂的应用程序，除了中央客户区界面，还可以包括菜单栏、工具栏、状态栏以及多个可停靠的工具对话框等等。
QWidget	最简单最基本的窗体程序，里面可以放置多个控件实现程序功能。
QDialog	基于对话框的程序，对话框一般用于弹窗，也可以用于主界面显示。对话框是从 QWidget 继承而来的，并丰富了一些功能，如模态显示和返回值等。

我们当然从最简单的学起，在基类里选择 QWidget，类名和文件名会根据基类自动修改，不需要额外设置。点击“下一步”，进入“项目管理”界面：



在项目管理界面可以设置作为子项目，以及加入版本控制系统管理。这两个功能暂时用不到，都用默认的，然后点击“完成”。

项目创建完成之后，Qt Creator 会直接进入代码编辑模式，可以看到类似下图界面：



1. 左边栏

编辑模式左边竖排的两个窗口叫做“边栏”，上面的默认是项目文件管理窗口，下面的是打开文件列表窗口。在 QtCreator 菜单“控件-->显示边栏 Alt+O”，可以控制边栏的显示和隐藏。

边栏里的窗口数目可以增加，边栏子窗口标题栏（其实是工具条，长得像标题栏，姑且这么称呼）有一排小按钮，最右边的是关闭按钮，倒数第二个是增加分栏按钮，可以添加多个边栏子窗口。

边栏子窗口标题栏第一个控件是组合框，可以选择该子窗口的功能视图类型，目前可以选择 8 个视图类型：

视图类型	说明
项目	即项目文件管理视图，可以选择项目里的文件进行编辑，包括 pro 文件也可以手动编辑。
打开文档	当前已经打开的文件列表，文件名右边如果有 * 号，是该文件被修改了但尚未保存。
书签	右击代码编辑器行号位置，看到“切换书签”，可以给代码行添加书签，方便跳转到该位置。
文件系统	相当于系统里的文件资源管理器，可以查看项目文件夹在磁盘里的实际文件列表。
类视图	可以查看项目里包含的类及相应源代码文件里的成员函数、成员变量。

视图类型	说明
大纲	编辑器所显示的当前文件的大纲列表，如名字空间、类名、成员函数、成员变量等。
类型层次	当前项目包含的类及其基类、派生类列表。
Include Hierarchy	包含视图，显示当前项目里 .h、.cpp 以及 Qt 类库头文件之间的包含关系。

可见 Qt Creator 提供的功能视图是很丰富的，这些视图不需要死记硬背，只要知道大概有这些东西，以后需要的时候会调出来就行了。一般用头两个就差不多了，当然也可以建立多个分栏，启用其他功能视图。

2. 代码编辑器

说完了左边栏，再来看看右边的代码编辑器：



代码编辑器大致可以分为三个部分，带有一堆控件的标题栏（其实是工具条）、行首区和编辑区。

先看看标题栏的 10 个控件，这些东西不需要记，因为打开 Qt Creator 的时候，这些东西都在那里，用鼠标指向这些控件几秒钟，会自动显示这些控件的工具提示信息，这里将它们罗列出来，方便读者以后查阅而已：

- ①和②：导航按钮“返回”和“前进”，这与网页浏览器的前进和后退按钮类似，可以在之前浏览的多个代码文件或一个代码文件里多个位置之间快速切换。
- ③：标识当前显示的文件是只读还是可写，一般都是可写的。
- ④：文件类型图标，当前显示文件的类型，这个控件其实是一个菜单按钮，点击可以弹出丰富的文件处理功能菜单，感兴趣的读者可以点开看看。

- ⑤：打开的文件名，可以在多个打开的文件之间选择切换，与边栏的“打开文档”视图是对应的。
- ⑥：关闭当前显示的文档。
- ⑦：为当前显示的文件添加额外的C++预处理指令，一般用不着。
- ⑧：选择符号，可以在当前显示的文件里多个函数、类、成员变量等之前快速切换，与边栏“大纲”视图是对应的。
- ⑨：编辑区光标的行号和列号。
- ⑩：代码编辑区分栏，可以增加多个编辑器窗口，显示多个打开的文档或显示较大源码文件的多个位置。

行首区是浅灰色背景的部分，主要用来显示代码行号，以及调试断点标志和代码书签标志。右击行首区可以弹出右键菜单，菜单里可以切换书签、编辑书签以及设置或取消断点。

同一行是既可以打断点也可以设置书签的，二者不冲突，其实它们根本就没关系。单击行号前面的浅灰色空白区可以直接打断点，再次单击可以取消断点，另外也可以用快捷键 F9 设置或取消断点。代码书签一般用右键菜单来设置，也可以用快捷键 Ctrl+M 设置或取消书签。

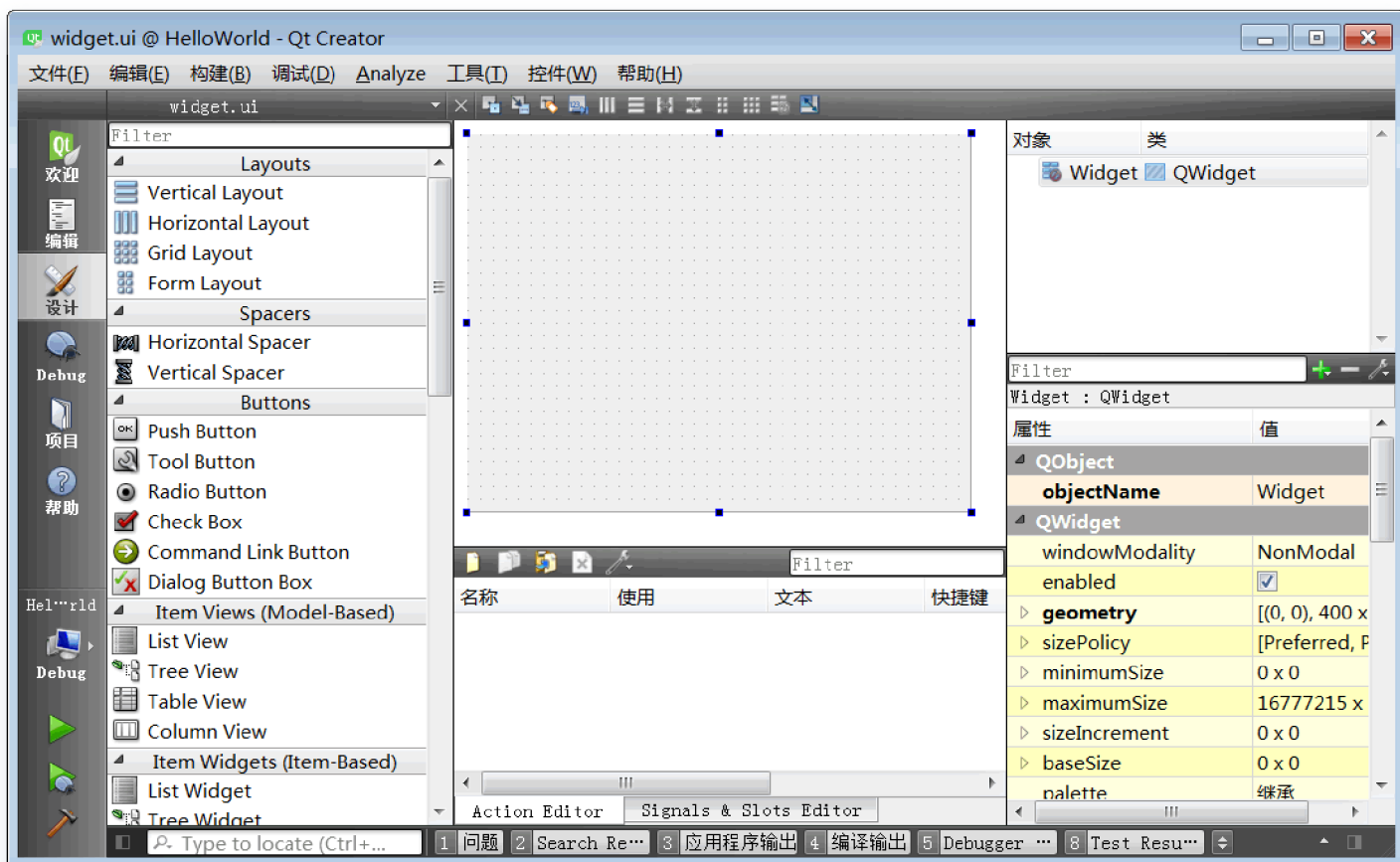
编辑区是程序员最为常用的部分了，就是写代码用的。编辑区当然有语法高亮显示了，而且从编辑区的复制出来的内容是 HTML 语法的丰富文本格式，如果粘贴到 Word 之类的文字处理软件中，会直接显示彩色高亮代码，这是很实用的功能。

对于现代集成开发环境常见的变量名、类名、名字空间、函数名、类对象成员变量、结构体成员变量等等名字补全功能，Qt Creator 编辑器当然也能很好地支持。变量/函数调用追踪、变量/函数声明追踪、类名或变量名函数名自动改名等常见的辅助功能，也都是支持的（选择要改或要追踪的名字，右击，在右键菜单里有一大堆功能，Refactor 菜单项里面有自动改名）。

1.3.4 Qt Creator 编辑 UI

在 Qt Creator 编辑模式边栏上面的项目视图里，包含一个 HelloWorld.pro 项目文件和 Headers（头文件）、Sources（源文件）、Forms（界面文件）三个虚拟目录。这三个虚拟目录是项目里对文件类型的归类显示，widget.h 和 main.cpp、widget.cpp 三个代码文件在后续小节讲解，本小节先设计图形界面文件 widget.ui。

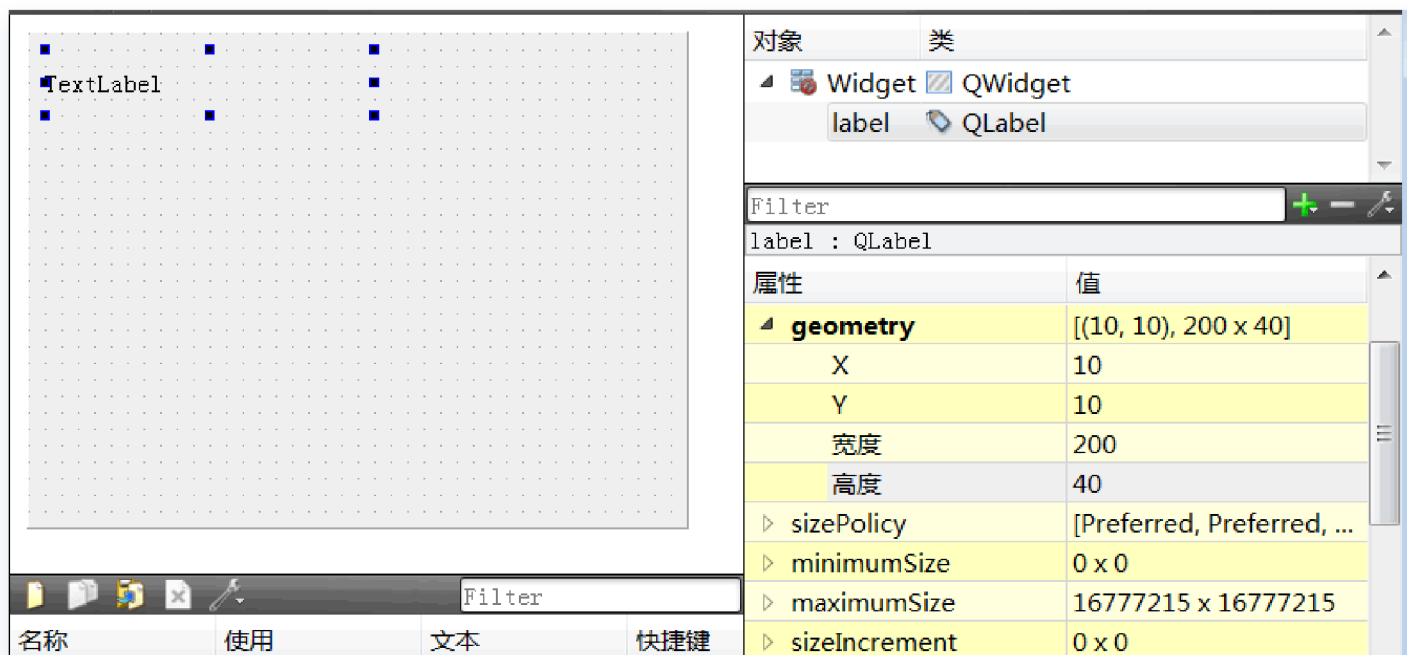
双击 widget.ui 文件，Qt Creator 会自动进入[设计模式](#)，可以对图形界面进行可视化编辑：



这个设计模式界面和《[Qt Designer的简单使用](#)》中介绍的 Qt 设计师是完全类似的，而且这就是将 Qt Designer 的功能做成插件，集成到 Qt Creator 了。

当然设计师变成插件之后，和原来的独立设计师程序有区别，现在插件设计师自己的菜单集成到 Qt Creator 菜单栏的“工具-->Form Editor”级联菜单里。预览窗口需要点击菜单栏的“工具-->Form Editor-->预览”。

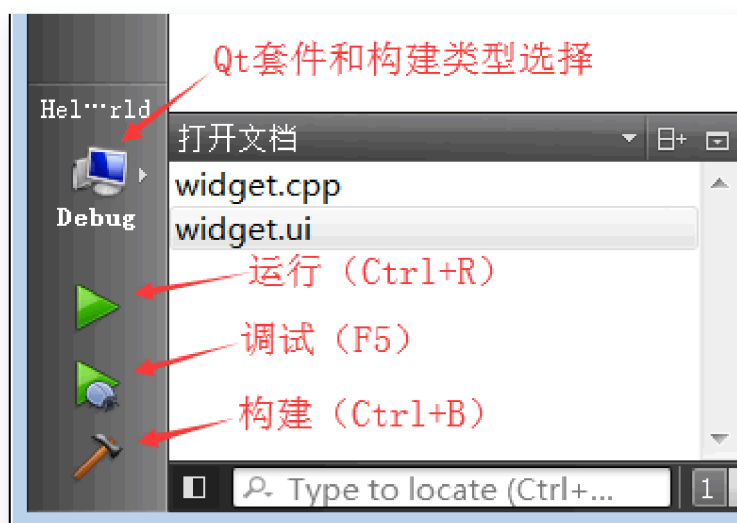
和《[Qt Designer的简单使用](#)》一节类似的，拖一个 Label 标签控件到窗体编辑区里，然后设置标签控件的 geometry 的四个子属性：X 为 10，Y 为 10，宽度为 200，高度为 40。接着编辑标签控件的 text 属性为“Hello World!”，看到效果如下：



编辑完成以后，注意使用 Ctrl+S 快捷键保存项目。

1.3.5 Qt Creator 生成和运行程序

使用 Qt Creator 生成和运行程序是再简单不过的事了。一个按钮就够了。当然，得先认识认识一下 Qt Creator 左下角的按钮：

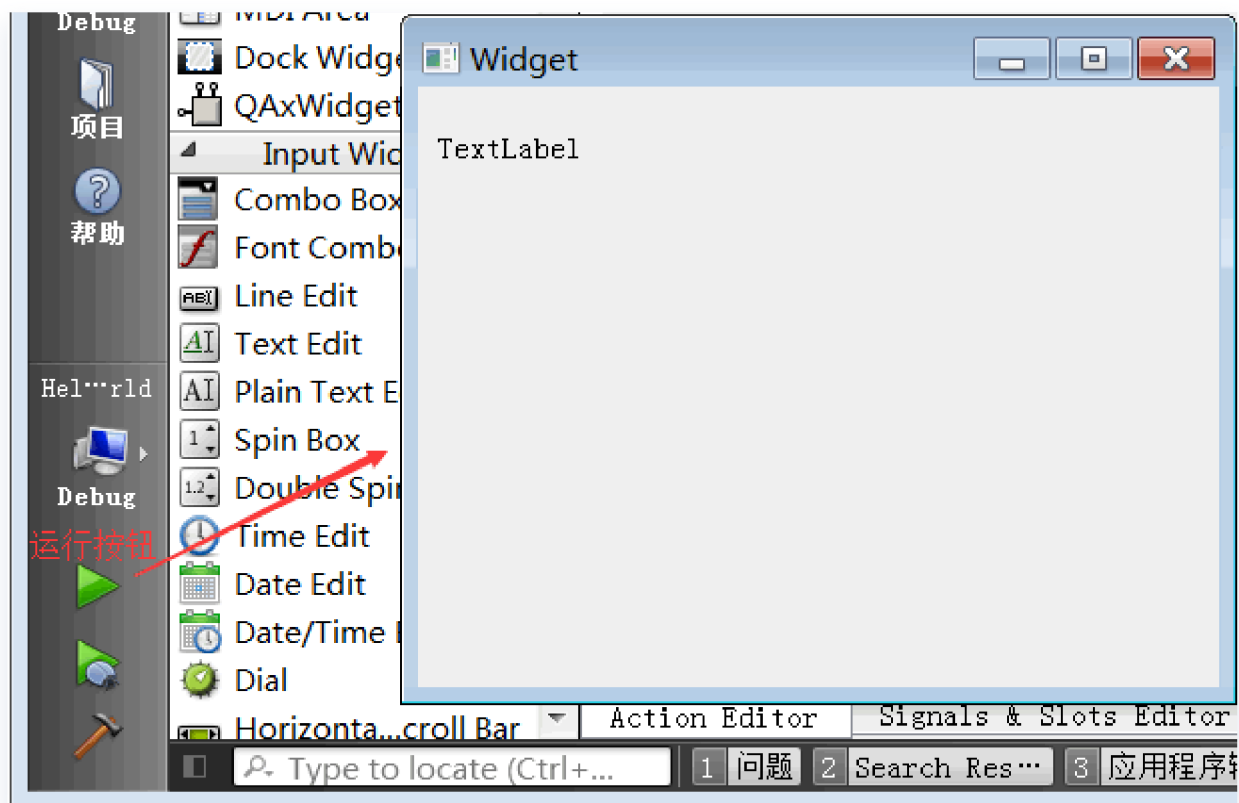


左下角一共有四个按钮，下面分别介绍一下：

- 第一个按钮是选择构建项目使用的 Qt 套件和构建目标程序的类型（Debug 或 Release）。
- 第二个是运行按钮，快捷键是 Ctrl+R，如果还没构建项目或刚修改了代码，直接点击运行的话，QtCreator 会自动构建生成新的目标程序并运行。
- 第三个是调试按钮，快捷键是 F5。调试程序之前，QtCreator 会自动构建生成最新的目标程序，并进入调试模式。在下一节专门讲解调试程序。

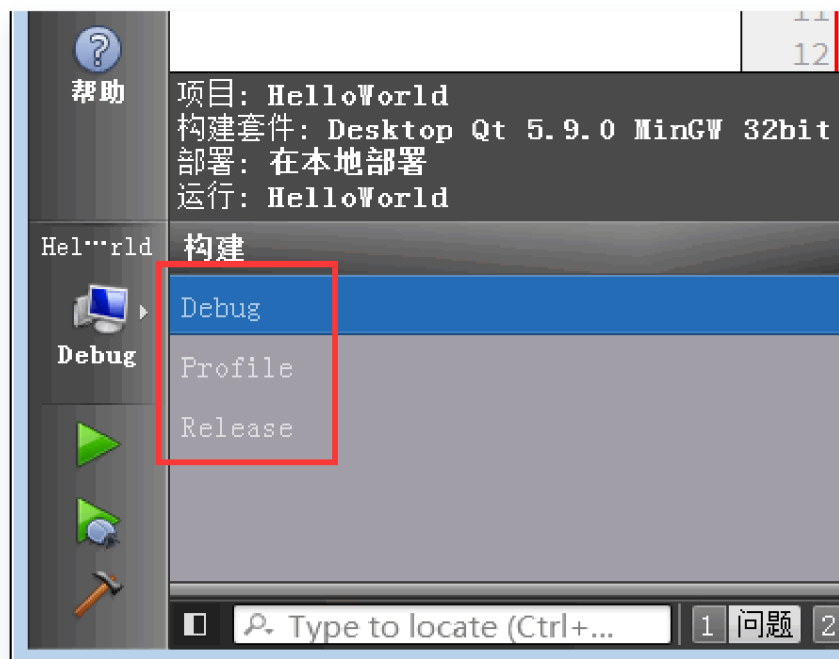
- 第四个是构建按钮，快捷键是 Ctrl+B，只构建最新的目标程序，但不运行。

如果只构建而不运行程序，就点第四个。一般都是构建后运行程序查看效果，可以直接点击第二个运行按钮，如果没问题发生，就会显示目标程序主界面：



从编译生成到运行，不需要自己敲命令，一个按钮搞定，这就是集成开发环境的好处。

上面示范的是默认 Qt 套件，构建的是 Debug 类型的目标程序。如果需要构建 Release 版目标程序，点开左下角第一个按钮：



这里有三种构建模式：其中 Debug 和 Release 我们都是耳熟能详的，前者是以 `-g` 模式编译，带着符号信息，便于我们调试，后者是经过优化之后，性能更上一个档次的。profile 则是在这两者之中取一个平衡，兼顾性能和调试，可以类似的看做是性能更优但是又方便调试的版本。

可以选择 Release 构建类型，然后再点击运行按钮就可以构建运行 Release 版本目标程序。

上图是针对项目只用到单一 Qt 套件的，如果之前配置了多个 Qt 套件，看到的类似下图：



如果项目配置了多个可用的 Qt 套件，点开左下角第一个按钮后，会看到各个套件以及构建类型，如果要切换 Qt 套件或构建类型，直接选中相应条目，然后点击运行按钮就行了。如果构建和运行时没出错，就会显示出构建好的目标程序界面。