

🔍 搜索

Qt创作者手册 > [运行自动测试](#)

# 运行自动测试

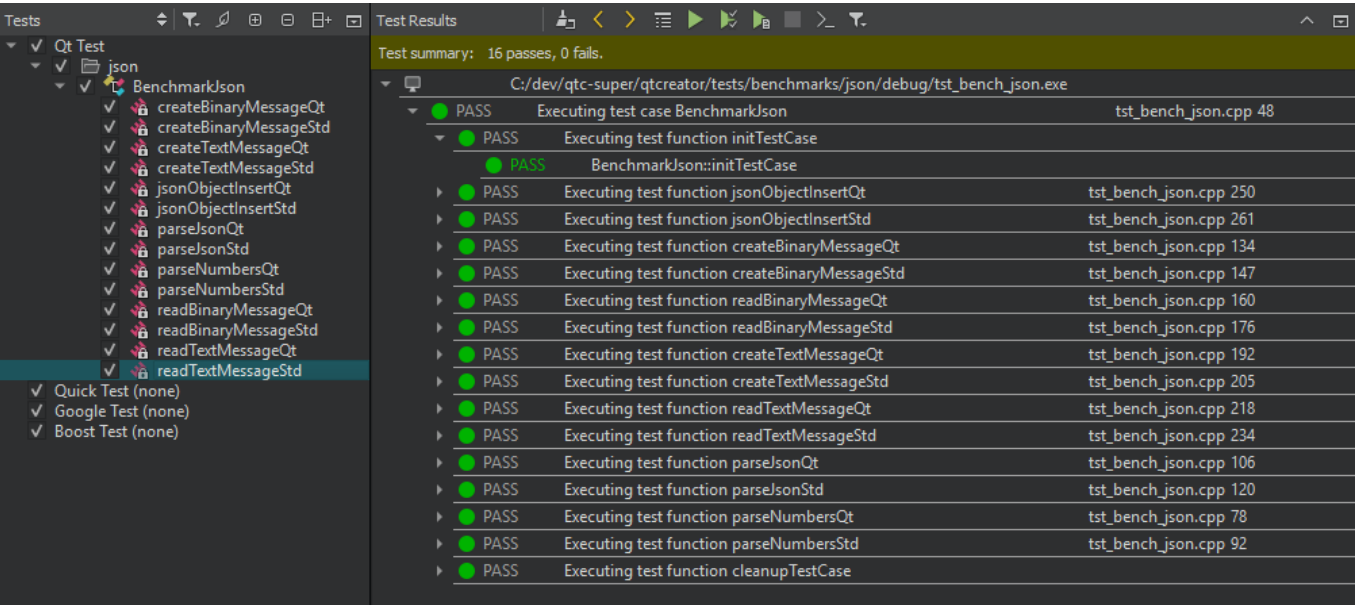
Qt Creator支持*基于代码的测试*和*基于构建系统的测试*。基于代码的测试为与底层代码模型或专用解析器紧密关联的特定测试框架提供特殊处理。基于构建系统的测试独立于任何测试框架。它直接从底层构建系统中检索信息，并使用它甚至构建系统来执行相应的测试。

Qt Creator为单元测试应用程序和库集成了以下测试框架：

- [提升测试](#)
- [Catch2 测试框架](#)
- [谷歌C++测试框架](#)
- [Qt测试框架](#)

为CTest 提供了额外的基于构建系统的支持。

您可以使用Qt Creator为您的项目创建、构建和运行基于代码的测试。



## 构建基于系统的测试

默认情况下，基于生成系统的测试的处理处于禁用状态，以避免干扰基于代码的分析器。要启用基于构建系统的测试，请在[首选项 > 测试 > 常规](#)中选择相应的测试工具。

如果同时启用了基于代码的测试和基于构建系统的测试，则可以在测试树中复制测试。另请参阅[选择要运行的测试](#)。

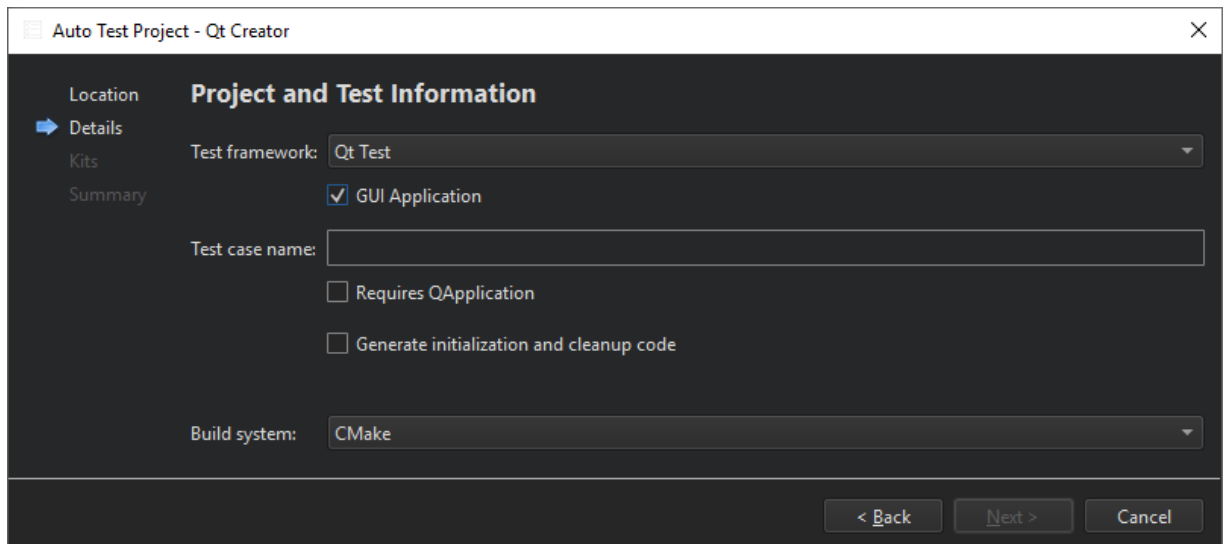
## 创建测试

可以使用向导创建包含测试的项目。

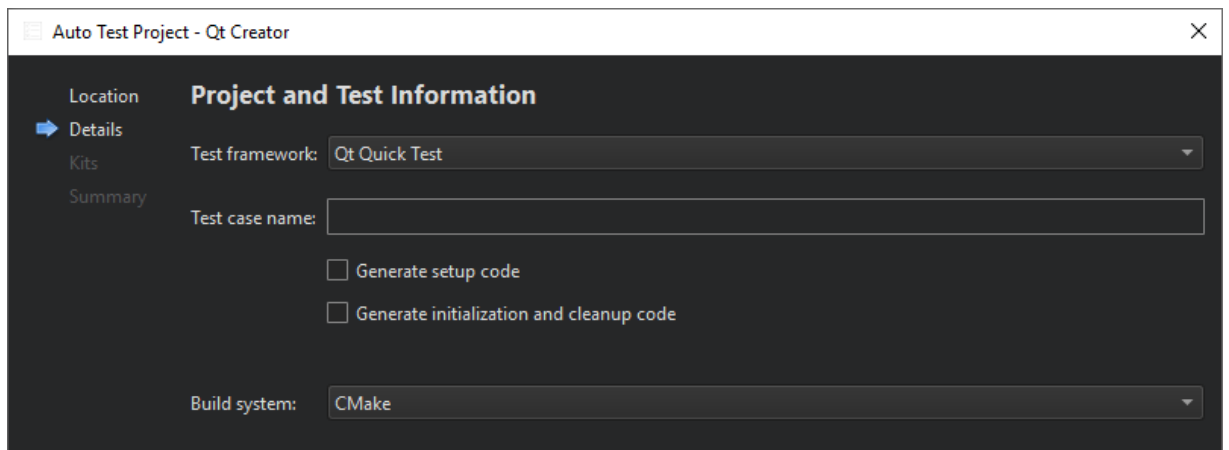
### 创建 Qt 和 Qt 快速测试

要创建 Qt 或 Qt 快速测试，请执行以下操作：

1. 选择**文件>新建项目>其他项目>自动测试项目>选择**使用样板代码创建用于Qt测试或Qt快速测试的项目。
2. 在“项目和测试信息”对话框中，指定**项目和测试**的设置：
  1. 在“**测试框架**”字段中，选择“Qt测试”或“Qt 快速测试”。
  2. 对于Qt测试，请选中GUI应用程序复选框以创建Qt应用程序。



3. 在 **测试用例名称** 字段中，输入测试用例的名称。
4. 对于 Qt 测试，请选中“**需要 QApplication**”复选框，将**QApplication**的 include 语句添加到项目的主.cpp 文件中。
5. 对于 Qt 快速测试，请选中**生成设置代码**复选框以在运行任何 QML 测试之前执行C++代码。测试框架将调用槽和可调用的函数，如在 [QML 测试之前执行C++](#)中所述。



6. 选中“**生成初始化和清理代码**”复选框，向测试添加由测试框架执行的函数，以初始化和清理测试。
7. 在“生成系统”字段中，选择要用于生成项目的**生成系统**：qmake、CMake 或 Qbs。

Qt Creator在指定的项目目录中创建测试。编辑.cpp文件，为测试中的每个测试函数添加专用槽。有关创建Qt测试的更多信息，请参阅[创建测试](#)。

## 创建谷歌测试

要创建 Google 测试，请执行以下操作：

1. 选择**文件>新建项目>其他项目>自动测试项目>选择**以使用 Google 测试创建包含样板代码的项目。
2. 在“项目和测试信息”对话框中，指定**项目和测试**的设置：
  1. 在“**测试框架**”字段中，选择“**Google 测试**”。
  2. 在 测试套件名称 字段中，输入**测试套件**的名称。
  3. 在 测试**用例名称**字段中，输入测试用例的名称。
  4. 选中启用**C++ 11**复选框以支持测试中的C++ 11 功能。
  5. 在“**Google 测试**存储库”字段中，选择包含 googletest 存储库克隆的目录。

要改用已安装的 Google C++ 测试框架，请参阅[设置 Google C++ 测试框架](#)。

6. 在“生成系统”字段中，选择要用于生成项目的**生成系统**：qmake、CMake 或 Qbs。

Qt Creator在指定的项目目录中创建测试。有关创建 Google 测试的更多信息，请参阅[Google 测试入门](#)。

## 创建提升测试

若要生成和运行 Boost 测试，必须在开发主机上安装 Boost.Test。通常，它是在安装Boost时安装的。您可以从[Boost.org](#) 下载Boost。

如果所使用的编译器和构建系统可以找到 Boost 库，则在创建测试时无需指定包含目录。

要创建提升测试，请执行以下操作：

1. 选择“**文件>新建项目>其他项目>自动测试项目>选择使用**样板代码创建用于 Boost 测试的项目。
2. 在“项目和测试信息”对话框中，指定**项目和测试**的设置：
  1. 在**测试框架**字段中，选择**提升测试**。
  2. 在 测试套件名称 字段中，输入**测试套件**的名称。
  3. 在 测试**用例名称**字段中，输入测试用例的名称。
  4. 在“**提升包含目录（可选）**”字段中，输入包含 Boost.Test 所需文件的目录的路径，例如`version.hpp`和包含测试头文件的名为`test`的子文件夹。
  5. 在“生成系统”字段中，选择要用于生成项目的**生成系统**：qmake、CMake 或 Qbs。

Qt Creator在指定的项目目录中创建测试。有关创建 Boost 测试的详细信息，请参阅[Boost.Test](#)。

## 创建 Catch2 测试

要构建和运行 Catch2 测试，您必须安装 Catch2 库和标头，或者您可以使用 Catch2 存储库提供的单个包含头文件。

如果所使用的编译器和构建系统可以自动找到 Catch2 标头，则在创建测试时无需指定包含目录。

1. 选择 **文件 > 新建项目 > 其他项目 > 自动测试项目 > 选择使用样板代码为 Catch2 测试创建项目。**

2. 在“项目和测试信息”对话框中，指定**项目和测试**的设置：

1. 在“**测试框架**”字段中，选择“**Catch2**”。
2. 在“**测试用例名称**”字段中，指定要用于**测试用例**文件的名称。
3. 选中“**使用 Qt 库**”复选框以使用自定义的主函数，并将项目设置为**使用 Qt 功能**。
4. 在 **Catch2 包含目录**（可选）字段中，您可以输入**包含 Catch2 头文件**的目录的路径。
5. 在“**生成系统**”字段中，选择要用于生成项目的**生成系统**：qmake、CMake 或 Qbs。

Qt Creator在指定的项目目录中创建测试。有关创建 Catch2 测试的详细信息，请参阅[Catch2](#)。

## 创建基于 CTest 的测试

CTest 提供为基于 CMake 的项目执行测试的功能，并且不限于特殊的测试框架。您只需在项目文件中配置测试，通常是 CMakeLists.txt。基本上，这是通过启用项目测试并注册测试应用程序甚至特殊命令来完成的。

```
enable_testing()
add_test(NAME test_example COMMAND test_example)
```

test\_example当然，在尝试将其注册为 test 之前，必须将其添加为可执行文件，或者它可以是任何命令，包括参数。有关如何使用 CTest 的详细信息，请参阅[使用 CTest 进行测试](#)。

## 设置 Google C++ 测试框架

要构建和运行 Google 测试，您必须在开发主机上安装和配置 Google C++测试框架。可以从 Git Hub 克隆它，也可以从安装包安装它。

要将项目配置为使用克隆的 Google 测试框架，请编辑项目文件（.pro）中的变量，以包含 Google Test'sand's 的源代码和文件夹。通常，您需要添加以下子文件夹：INCLUDEPATHincludegoogletestgooglemock

```
> googletest
> googlemock
> googletest/include
> googlemock/include
```

您还需要将必要的文件添加到变量中。例如：SOURCES

```
> googletest/src/gtest-all.cc
> googlemock/src/gmock-all.cc
```

要将项目配置为使用已安装的 Google 测试框架包，请将以下包含路径添加到 .pro 文件：

```
> <googletest_install_path>/include/gtest
> <googlemock_install_path>/include/gmock
```

然后添加链接器选项，以便能够找到库并链接到它们。例如，对于基于 qmake 的项目，通常需要将以下值添加到 .pro 文件中：

# 生成和运行测试

生成并运行测试：

1. 打开包含测试的项目。
2. 在“**测试**”视图中，选择要运行的测试。
3. 在“**测试结果**”中，选择：
  - ▶ (运行所有测试) 以运行所有测试。
  - ▶ (运行选定的测试) 以运行选定的测试。
  - ▶ (运行失败的测试) 以重新运行上次运行中失败的测试。根据框架的不同，如果无法区分或完全解决测试，这可能会选择其他测试。
  - ▶ (对当前文件运行测试) 以在代码编辑器中当前打开的文件中运行测试。

默认情况下，Qt Creator在部署和运行项目之前构建项目。若要运行所有测试而不生成并再次部署它们，请在上下文菜单中选择“**运行所有测试而不部署**”。若要在不部署的情况下运行所选测试，请选择“**在不部署的情况下运行所选测试**”。

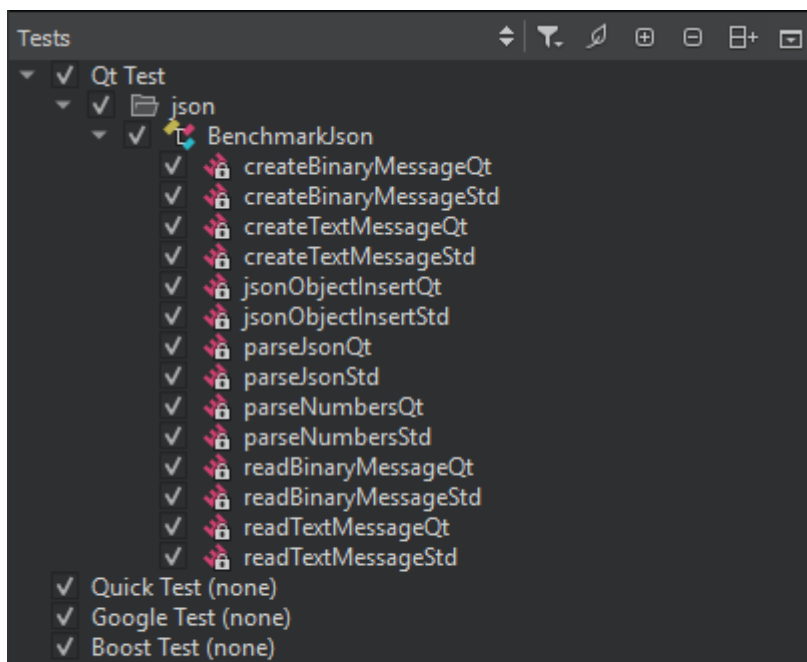
用于运行测试的函数也可以在“**测试**”视图的上下文菜单和“**工具>测试**”中使用。

**注意：**如果启用了基于生成系统和基于代码的测试，则在使用“运行所有测试”或“**运行所选测试**”时，可以**运行测试**两次。如果基于代码的测试框架可以找到测试，并在构建系统中注册为测试，则会发生这种情况。

如果测试的执行时间超过一分钟，则默认超时可能会停止测试执行。若要增加超时，请选择“**编辑>首选项>测试>常规**”。

## 选择要运行的测试


“测试”视图显示为当前项目中当前活动的测试框架找到的所有**测试**。选择要运行的测试用例。



如果Qt Quick测试用例没有名称，则会在列表中标记为**未命名**。选择“**运行所有测试**”时，将执行未命名的测试用例。您无法选择或取消选择它们。

Qt Creator会在您打开项目时扫描项目中的测试，并在编辑测试时更新当前活动的测试框架的测试列表。若要刷新视图，请在上下文菜单中选择“**重新扫描测试**”。

若要在“测试”视图中显示或隐藏初始化和清理或数据函数，请选择“（**筛选测试树**）”，然后选择“**显示初始化和清理函数**”或“**显示数据函数**”。双击列表中的函数以在代码编辑器中打开其源代码。

测试用例按字母顺序列出，不区分大小写。要按它们在源代码中定义的顺序列出它们，请选择  （**自然排序**）。

## 从代码编辑器运行和调试测试

可以在代码编辑器中当前打开的文件中运行和调试测试。若要运行打开文件中的所有测试，请选择“**工具>测试>运行当前文件的测试**”。

**注意：**仅适用于基于代码的测试框架。

若要运行当前在打开的文件中选定的测试，请在上下文菜单中的“**光标**”下选择“**运行测试**”。

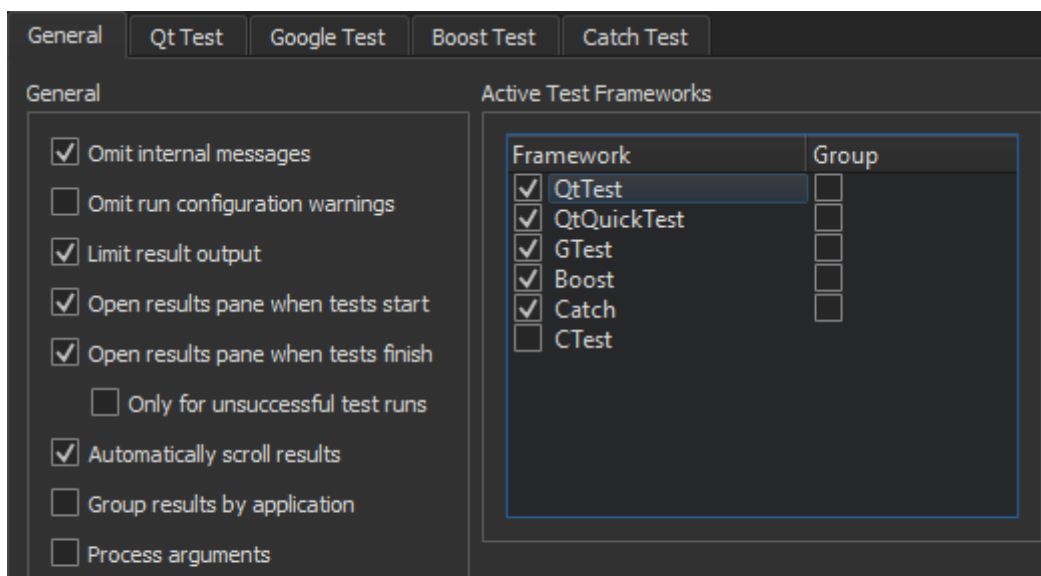
**注意：**仅适用于基于代码的测试框架。

若要调试当前选定的测试，请在上下文菜单中的“**光标**”下选择“**调试测试**”。

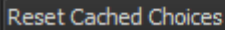
**注意：**基于生成系统的测试的可用性取决于生成系统工具的功能。

## 指定测试设置

若要自定义测试、测试框架和测试工具的处理，请选择“**编辑>首选项>测试>常规**”。







您可以在项目级别自定义某些设置。若要更改当前项目的设置而不是全局设置，请选择“**项目>项目设置>测试**”。

在“**活动测试框架**”列表中，您可以选择Qt Creator将处理的测试。若要提高测试的完全扫描的性能，请禁用不使用的测试框架。

若要对活动测试框架的相关测试用例进行分组，请选中“**活动测试框架**”列表中框架名称旁边的“**分组**”复选框。默认情况下，测试根据其所在的目录进行分组。

默认情况下，省略推导配置的内部消息和运行配置警告。要查看它们，请取消选中“**省略内部消息**”和“**省略运行配置警告**”复选框。

默认情况下，测试结果输出限制为 100,000 个字符。添加新结果时，视图会自动向下滚动。要显示完整结果，请取消选中“**限制结果输出**”复选框。要禁用自动滚动，请取消选中**自动滚动结果**复选框。

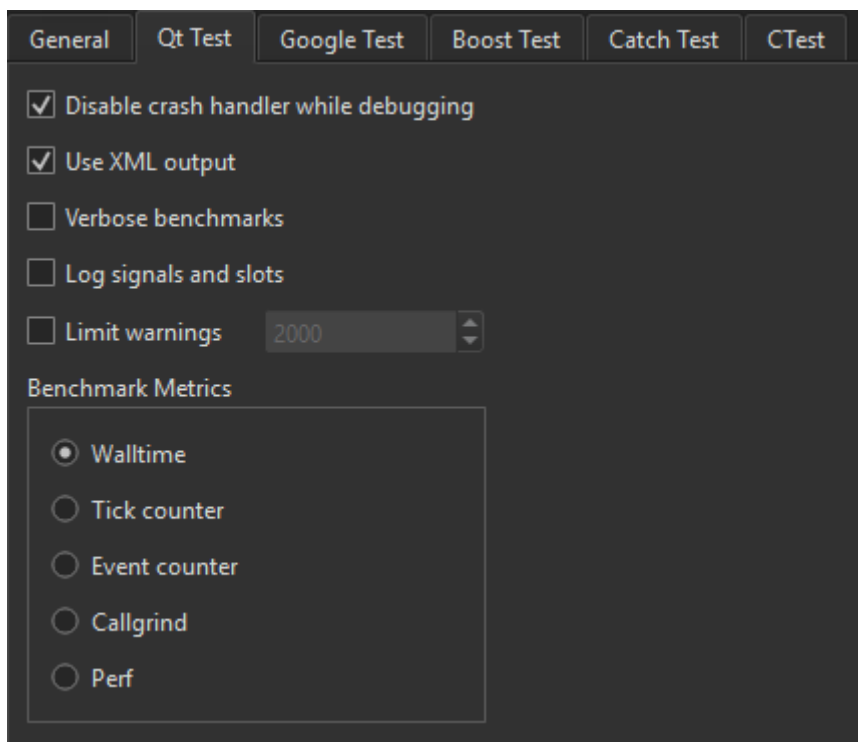
测试结果可以按用于运行测试的可执行路径进行分组。如果您有多个测试可执行文件并一次运行它们，这将非常有用。若要启用此功能，需要选中“**按应用程序对结果进行分组**”复选框。

成功生成当前项目后，可以自动运行当前可用的测试。在“**自动运行**”中，选择成功生成后应运行的测试。

在某些特殊设置中，Qt Creator无法推断出它应该使用哪个可执行文件或运行配置。如果Qt Creator在尝试执行测试时反复要求您选择要运行的测试，您可以启用它来缓存您的选择并使用它们。当您切换到另一个项目、关闭当前项目或选择“**重置缓存的选择**”时，缓存的信息将被清除。

## 指定运行Qt测试的设置

基准测试中的代码被测量，并且可能还会重复几次以获得准确的测量。这取决于您可以在 **Edit>Preferences>Testing>Qt Test** 的基准**指标**组中选择的测量后端：walltime、CPU tick 计数器、事件计数器、Valgrind Callgrind 和 Linux Perf。有关更多信息，请参阅[创建基准](#)。



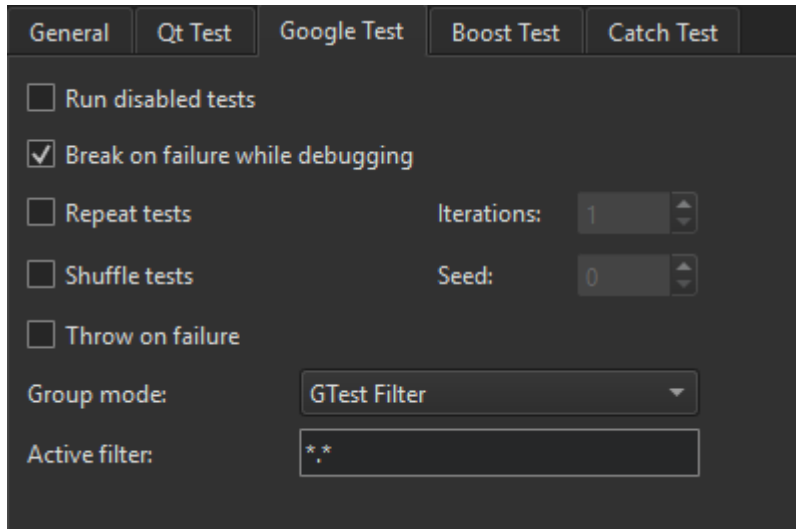
若要在运行基准时接收详细输出，请选中“**详细基准**”复选框。

若要禁用测试器中显示上的 Qt 测试，请选中“**测试时禁用输出处理程序**”复选框。

若要显式限制测试日志中的最大警告数，请选中“**限制警告**”复选框，并在旁边的数字显示框中指定所需的数量。如果您根本不要限制，请将数字设置为 0。默认数字为 2000。

## 指定运行 Google 测试的设置

要指定运行 Google 测试的设置，请选择**编辑>首选项>测试>Google 测试**。



若要运行禁用的测试，请选中“**运行禁用的测试**”复选框。

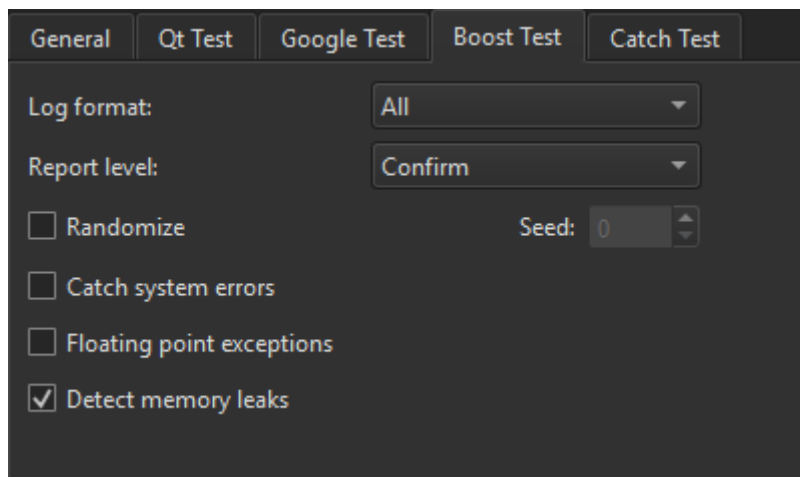
若要运行测试的多个迭代，请选中“重复测试”复选框，然后在“**迭代**”字段中输入应运行**测试**的次数。若要确保测试是独立且可重复的，可以通过选中“**随机测试**”复选框，每次以不同的顺序运行它们。

若要将故障转换为调试器断点，请选中“**调试时失败时中断**”复选框。若要将断言失败转换为 C++ 异常，请选中“**失败时抛出**”复选框。

要使用 GTest 筛选器对 Google 测试进行分组，请在“**分组模式**”字段中选择“**GTest 筛选器**”，然后在“**活动筛选器**”字段中指定要使用的筛选器。有关 GTest 筛选器的详细信息，请参阅[运行测试的子集](#)。

## 指定运行提升测试的设置

1. 若要指定运行提升测试的设置，请选择“**编辑>首选项>测试>提升测试**”。



2. 在“**日志格式**”字段中，选择错误报告格式以指定要在测试报告中记录的事件类型。
3. 在 **报告级别** 字段中，选择测试结果**报告的详细级别**。如果不需要报告，请选择“**否**”。

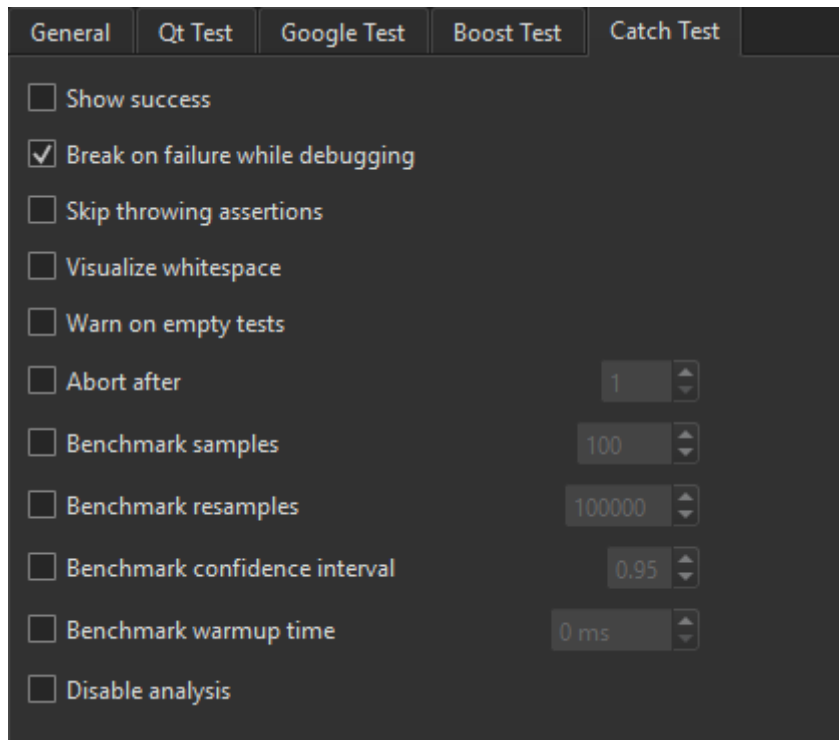


8. 选中“**行点并市**”复选框以**检测内存泄漏**。

7. 选中“检测内存泄漏”复选框以**检测内存泄漏**。

## 指定运行 Catch2 测试的设置

1. 若要指定运行 Catch2 测试的设置，请选择“**编辑>首选项>测试>捕获测试**”。



2. 选中“显示成功”复选框以**显示**后续表达式。默认情况下，Catch2 只会打印失败。

3. 选中“调试时**失败时中断**”复选框，将故障转换为调试器断点。

4. 选中“跳过引发断言”复选框以跳过测试引发异常的任何**断言**。

5. 选中“**可视化空白**”复选框可将空格转换为转义序列。

6. 选中“**空**测试时发出警告”复选框，以便在测试用例未选中任何断言时收到警告。

7. 选中“**在此时间**后中止”复选框，在数字显示框内指定的失败次数后中止测试。

8. 选中基准样本复选框以指定运行基准时要收集的**样本数**。

9. 选中基准重采样复选框以指定在基准测试后用于统计自举的**重采样数**。

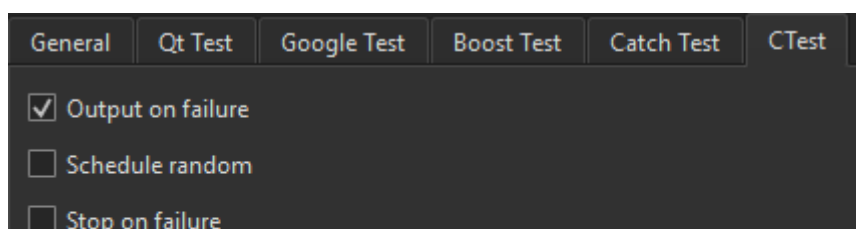
10. 选中基准置信区间复选框以指定用于统计引导的**置信区间**。

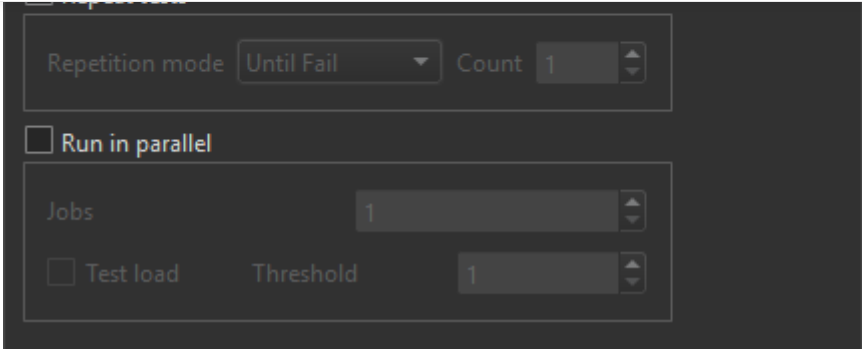
11. 选中“基准测试预热时间”复选框以指定基准测试开始之前每个测试的**预热时间**。

Qt创建者手册8.0.2  
Topics >

## 指定运行基于 CTest 的测试的设置

1. 若要指定运行基于 CTest 的测试的设置，请选择“**编辑>首选项>测试>CTest**”。





- 2. 选中“失败时输出”复选框，以便在测试**失败时**显示特定于测试的输出。与 CTest 默认值相反，默认情况下启用此功能。
- 3. 选择“**随机计划**”以随机顺序执行测试。
- 4. 选择“失败时停止”，在第一个失败的测试**时自动停止**测试执行。
- 5. 在**输出模式**字段中，选择 CTest 输出的详细级别。

**注意：**这只会影响文本显示上的输出。

- 6. 如果要在某些情况下重新运行测试，请选择“**重复测试**”。
- 7. 在 重复模式 字段中，选择重新运行测试的**模式**。可以在“计数”字段中指定重复测试的最大**计数**。
- 8. 选择“并行运行”，使用指定数量的**作业并行运行**测试。
- 9. 选择“**测试负载**”以能够限制并行执行。如果 CTest 会导致 CPU 负载超过阈值中给出的**阈值**，则不会启动新测试。

## 查看测试输出

测试结果以 XML 格式显示在**测试结果**中。XML 可以比纯文本更轻松、更可靠地解析。

但是，如果Qt测试崩溃，它可能无法生成可解析的完整XML代码，这可能会导致信息丢失。以纯文本形式查看结果时，丢失的信息可能可检索。若要以纯文本形式查看 Qt 测试的结果，请选择“编辑>**首选项**>测试>Qt **测试**”，然后取消选中“**使用 XML 输出**”复选框。然后选择“测试结果”中的 **>-**（在**可视显示**和**文本显示**之间切换）按钮以切换到文本显示。


下表列出了**测试结果**显示的消息：

结果	描述
板凳	基准测试。
调试	调试消息。
失败	测试用例失败。双击该行以获取更多信息。
致命	例如，发生了阻止运行测试用例的致命错误。
信息	信息性消息。
内部	内部消息。
通过	测试用例通过。
跳	跳过测试用例。

	(XPASS) 写入测试日志。
XPASS	测试用例通过了，即使它预计会失败。
警告	警告消息。

从Qt 5.4开始，您可以提供黑名单文件进行测试。它主要由Qt CI系统内部使用。

结果	描述
布菲尔	列入黑名单的测试用例失败。
巴帕斯	列入黑名单的测试用例已通过。
BXFAIL和BXFAIL	列入黑名单的测试用例失败，但被标记为预期失败。
BXPASS	列入黑名单的测试用例通过了，即使预计会失败。

若要仅查看特定类型的邮件，请选择“（筛选测试结果）”，然后选择要显示的邮件类型。若要显示所有邮件，请选择“检查所有筛选器”。要取消选择所有邮件类型，请选择取消选中所有过滤器。

< 可视化 Chrome 跟踪事件

高级使用 >

©2022 Qt有限公司 此处包含的文档贡献的版权归 他们各自的所有者。此处提供的文档根据自由软件基金会发布的GNU自由文档许可证版本 1.3的条款进行许可。Qt和相应的徽标是Qt有限公司在芬兰和/或全球其他国家的商标。所有其他商标均为财产 其各自所有者。



联系我们

公司

- 关于我们
- 投资者
- 编辑部
- 职业
- 办公地点

发牌

- 条款和条件
- 开源
- 常见问题

支持

- 支持服务
- 专业服务

对于客户

- 支持中心
- 下载



社区

- 为Qt做贡献
- 论坛
- 维基
- 下载
- 市场

© 2022 Qt公司

[反馈](#) [登录](#)