

分析应用程序

在这个部分：

- [依赖关系分析](#)
- [使用DSM分析相关性](#)
- [查看源代码的结构和层次结构](#)
- [分析反向依赖关系](#)
- [分析循环依赖关系](#)
- [分析依赖关系](#)
- [分析重复](#)
- [分析模块依赖关系](#)
- [分析外部堆栈跟踪](#)
- [分析数据流](#)
- [验证依赖关系](#)

依赖关系分析

在这个部分：

- 用法分析，可帮助您找到对某个类，变量，方法或参数的所有引用。该工具包括[搜索](#)和[查看](#)整个项目的[用法](#)，并[突出显示](#)文件中的[用法](#)。
- 可以[查看文件结构](#)。
- 可以[探索](#)类型，方法和方法调用的[层次结构](#)。
- 搜索[重复的代码片段](#)。
- 依赖性分析（[模块](#)，[向后](#)或[循环](#)）。
- 使用[依赖结构矩阵分析](#)探索复杂的依赖[关系](#)。

最后修改日期：2017年12月22日

使用**DSM**分析相关性

此功能仅在**Ultimate**版本中受支持。

此功能仅支持**Java**！

在这个部分：

- 使用DSM分析相关性
- [DSM分析](#)
- [访问DSM分析](#)
- [扩展依赖关系](#)
- [探索依赖关系](#)
- [查找相关性的用法](#)
- [限制DSM范围](#)

IntelliJ IDEA通过捆绑插件实现DSM分析功能，通过清除 IntelliJ IDEA设置（）的插件页面上的DSM分析复选框，可以完全禁用该 插件 **Ctrl+Alt+S**。

最后修改日期：2017年12月22日

DSM分析

此功能仅在**Ultimate**版本中受支持。

此功能仅支持**Java**！

在这个部分：帝斯曼帮助。

DSM代表依赖结构矩阵（**Dependency Structure Matrix**） - 一种探索程序部分（模块，类别等）之间的依赖关系的方法，并提供项目的紧凑矩阵表示。

DSM分析帮助您可视化项目各个部分（模块，类等）之间的依存关系，并突出显示项目中的信息流。

DSM分析可以用来管理变化如何影响项目。例如，如果需要更改其中一个类，则可以识别所有依赖关系，并查看更改将如何在项目中传播。依赖关系结构矩阵列出项目的所有部分以及它们之间的依赖关系。

DSM分析的结果显示在特殊的[DSM视图中](#)。

只有在IntelliJ IDEA安装中启用了**DSM Analysis**插件后才能使用此功能。

最后修改日期：2017年12月22日

访问**DSM**分析

此功能仅在**Ultimate**版本中受支持。

此功能仅支持**Java**！

调用**DSM**

1. 在主菜单上，选择分析 | 分析依赖矩阵。请注意，如果只启用了**DSM**分析插件，则此命令在菜单上可用。请参阅设置对话框的[插件设置](#)页面。
2. 在“指定分析依赖性矩阵范围”对话框中，单击与所需分析范围对应的单选按钮。
3. 如果要分析测试来源，请选中包含测试来源复选框。点击**OK**。
如果您的项目类文件已过时，分析可能会导致数据不完整或不正确。为了避免这种情况，IntelliJ IDEA在继续进行**DSM**分析之前询问您是否要编译项目。
4. 准备就绪后，**DSM View**将在新窗口中打开，使您可以检查依赖关系。

最后修改日期：2017年12月22日

扩展依赖关系

此功能仅在**Ultimate**版本中受支持。

此功能仅支持**Java**！

您可以扩展依赖关系来更详细地检查它们。

扩展依赖关系

1. 在DSM视图中，选择要浏览的单元格。
2. 双击该单元格。这将更详细地展开显示依赖关系的两行。

最后修改日期：2017年12月22日

探索依赖关系

此功能仅在**Ultimate**版本中受支持。

此功能仅支持**Java**！

您可以扩展依赖关系来更详细地检查它们。限制范围，只剩下产生依赖关系的类。

扩展依赖关系

1. 在DSM视图中，选择要浏览的单元格。
2. 双击该单元格。这将更详细地展开显示依赖关系的两行。在上下文菜单中探索依赖关系。产生这些依赖关系的类将在DSM视图中的新选项卡中打开。

最后修改日期：2017年12月22日

查找相关性的用法

此功能仅在**Ultimate**版本中受支持。

此功能仅支持**Java**！

您可以在[Find Usages视图](#)中打开选定的依赖关系，以进一步进行源代码分析。

找到依赖的用法

1. 在DSM视图中，选择相关性。
2. 右键单击依赖项，然后在上下文菜单上选择“查找用于依赖关系”。

最后修改日期：2017年12月22日

限制**DSM**范围

此功能仅在**Ultimate**版本中受支持。

此功能仅支持**Java**！

您可以将**DSM**的范围限制为选定的行。只有这些才会留在新的矩阵中。

限制**DSM**范围

1. 在**DSM**视图中，选择要保留的行。
2. 右键单击选定的行并在菜单中选择“限制范围选择”。有限的范围将在**DSM**视图的新标签中打开。

最后修改日期：2017年12月22日

查看源代码的结构和层次结构

IntelliJ IDEA使您能够在“层次结构”工具窗口中检查类，方法和调用的层次结构，并在“结构”工具窗口中浏览源文件的结构。

- “视图”菜单提供了“层次结构”和“结构”工具窗口。
- 该层次工具窗口才可以用，当一个层次是建立。
- 层次结构在“导航”菜单中构建。

您可以将DSM的范围限制为选定的行。只有这些才会留在新的矩阵中。

- [构建调用层次结构](#)
- [建筑类层次结构](#)
- [构建方法层次结构](#)
- [保留层次结构选项卡](#)
- [查看层次结构](#)
- [查看源文件的结构](#)

构建调用层次结构

您可以在[层次结构工具窗口中](#)构建和查看所选方法的调用者和被调用者的[层次结构](#)。在查看呼叫层次之前，您需要至少构建一个。

构建方法调用的层次结构

1. 在编辑器中，将插入符号放在方法声明或用法上。在项目视图或其他工具窗口中，选择所需的方法。
2. 执行以下任一操作：
 - 在主菜单上，选择导航 | 调用层次结构。
 - 按 `Ctrl+Alt+H`。

最后修改日期：2017年12月22日

建筑类层次结构

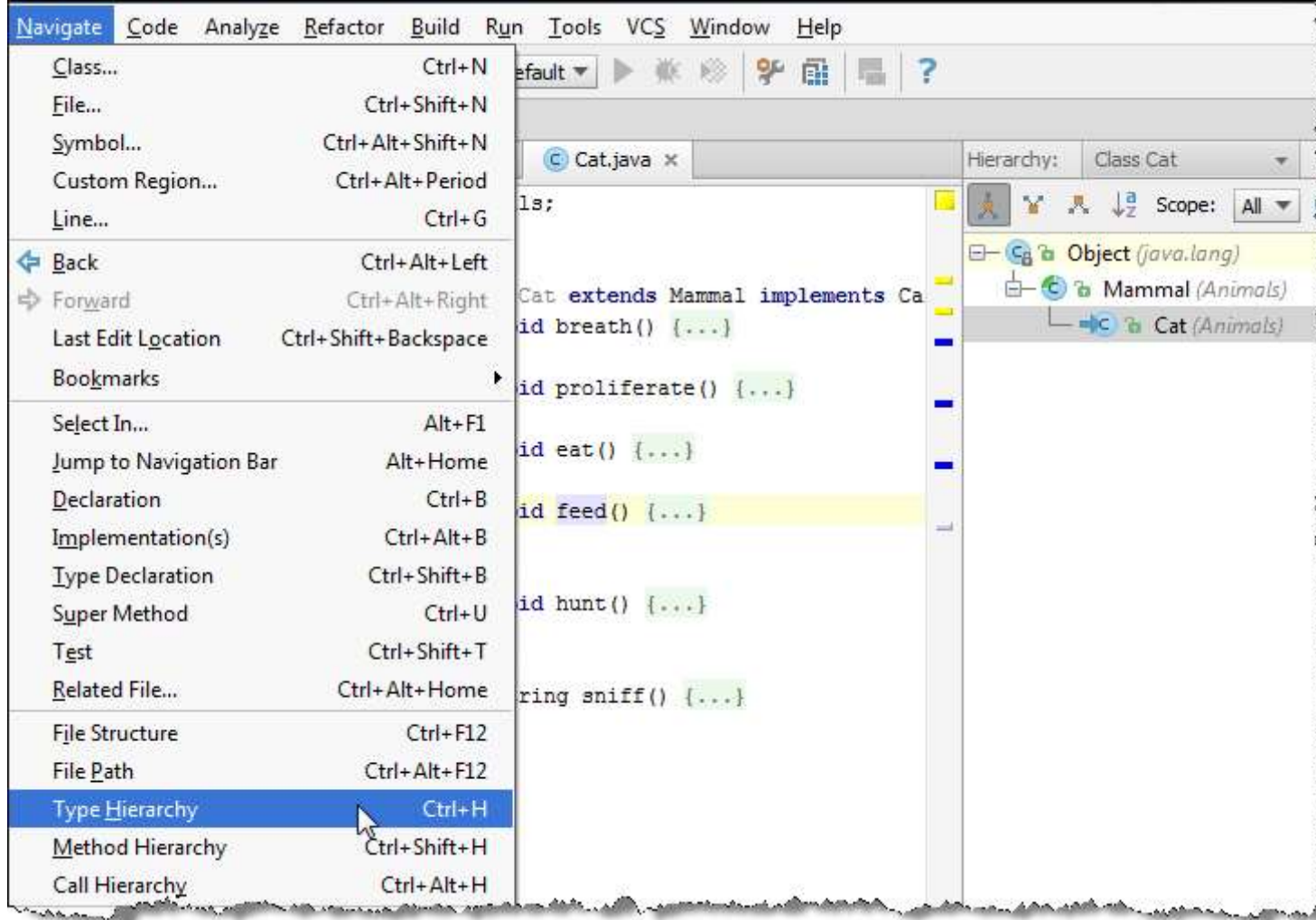
您可以在[层次结构工具窗口](#)中构建和查看所选方法的调用者和被调用者的[层次结构](#)。在查看呼叫层次之前，您需要至少构建一个。

请注意，如下所述，构建类层次结构时，只能使用层次结构工具窗口。

构建类的层次结构

1. 在“项目”工具窗口中选择所需的类，或者在编辑器中打开它。
2. 在主菜单上，选择导航 | 键入层次结构或按 **Ctrl+H**。

类层次结构显示在“[层次结构](#)”工具窗口中。



Next
构建类层次

Previous
结构和层次最后修改日期: 2017年12月22日

构建方法层次结构

方法层次结构可以检查给定方法的类的树视图：

- defined (+) 。
- 没有定义，但在超类 (-) 中定义。
- 被定义，因为这个类不是abstract (!) 。

构建方法层次结构

1. 在“项目”工具窗口中选择所需的方法，或将脱字号放在编辑器中的方法声明中。
2. 执行以下任一操作：
 - 在主菜单上，选择导航 | 方法层次结构。
 - 按 `Ctrl+Shift+H` 。

最后修改日期：2017年12月22日

保留层次结构选项卡

默认情况下，每次构建新的层次结构时，IntelliJ IDEA将覆盖[Hierarchy](#)工具窗口中的当前选项卡。您可以保留所需选项卡的内容，并在新选项卡中构建下一个层次结构。

保留层次结构选项卡

- 在[Hierarchy](#)工具窗口中，单击工具栏上的**Pin**标签按钮.

最后修改日期：2017年12月22日

查看层次结构

默认情况下，每次构建新的层次结构时，IntelliJ IDEA将覆盖[Hierarchy](#)工具窗口中的当前选项卡。您可以保留所需选项卡的内容，并在新选项卡中构建下一个层次结构。[层次结构工具窗口](#)。

在本页：

显示层次结构工具窗口

没有要显示的层次结构时，不显示层次结构工具窗口。你必须先建立层次结构。

请参阅[构建类层次结构](#)，[构建调用层次结构](#)和[构建方法层次结构](#) 以了解如何构建层次结构。

要打开**Hierarchy**工具窗口，请执行以下操作之一

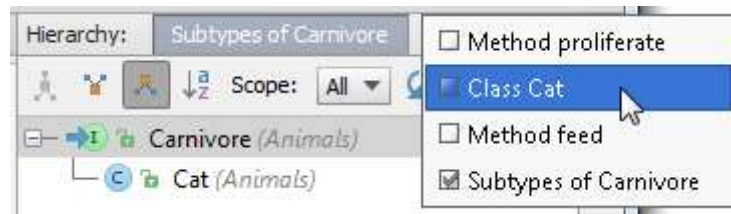
- 在主菜单上，选择查看 | 工具窗口 | 层次结构。
- 使用 **Alt+8** 键盘快捷键。

在“层次结构”工具窗口的选项卡之间导航

执行以下任一操作：



- 右键单击当前显示的选项卡，然后选择 上下文菜单上的选择下一个选项卡 / 选择上一个选项卡。

- 使用 **Alt+Right** 和 **Alt+Left** 键盘快捷键。
- 单击当前显示的选项卡，然后选择要显示的下一个选项卡。



在视图之间切换

切换视图意味着显示升序或降序的层次结构（被调用者与调用者方法，父对子类等）要切换视图，请使用层次结构工具窗口的工具栏：

- 单击  以显示调用方法或超类型。
- 单击  显示被调用的方法或子类型。

最后修改日期：2017年12月22日

查看源文件的结构

在本页：

- [基本](#)
- [查看文件的结构](#)
- [查看成员](#)

基本

您可以使用“结构”工具窗口或“结构”弹出窗口来检查当前在编辑器中打开的文件的结构。


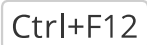
默认情况下，IntelliJ IDEA显示当前文件中提供的所有类，方法等。

要[显示其他成员](#)，请单击[结构](#)工具窗口工具栏上的相应按钮。

您还可以在“[项目](#)”工具窗口中显示班级成员。

查看文件的结构

要查看文件结构，请执行以下操作之一

- 在主菜单上，选择查看 | 工具窗口 | 结构，显示 [结构工具窗口](#)。
- 按结构工具按钮显示 [结构工具窗口](#)。
- 按下  显示 [结构工具窗口](#)。
- 按  显示 [结构弹出](#)。

查看成员

要显示类字段


- 点击  结构工具窗口的工具栏。

要显示继承的成员

- 点击  结构工具窗口的工具栏。

默认情况下，IntelliJ IDEA 只显示当前类中定义的方法，常量和字段。如果显示，继承的成员显示为灰色。

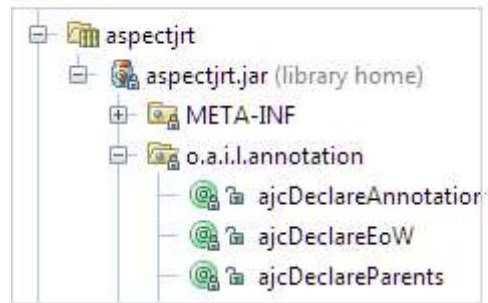
显示包含的文件

- 点击  工具栏上的。

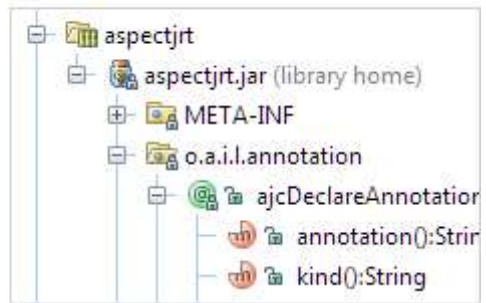
要在“项目”工具窗口中显示类成员

- 打开项目 工具窗口标题栏的上下文菜单上的显示成员项目。如果启用此选项，树中包含类的文件将变成节点。当展开这样的节点时，将显示包含它们的字段，方法和所选项目的其他成员的类。

Show Members: OFF



Show Members: ON



最后修改日期: 2017年12月22日

分析反向依赖关系

通过这种类型的分析，您可以在某个特定范围内找到另一个类或模块，这取决于指定的分析范围（整个项目，模块，文件，未版本化的文件等）。分析结果显示在[Dependency Viewer](#)的专用选项卡中。

后向依赖分析可能会相当耗时，特别是在大型项目上。

分析项目的后向依赖关系

1. 在主菜单上，选择 分析 | 分析反向依赖关系。“[指定后向相关性分析范围](#)”对话框打开。
2. 在“分析范围”部分中，指定您想要查找依赖关系的项目部分。
3. 在“感兴趣的范围”部分中，指定要查找依赖关系的范围。您可以从下拉列表中选择一个预定义的范围，或者单击省略号按钮，并在“[范围](#)”对话框中创建您自己的范围。
4. 如果您想分析测试来源，请 选择包含测试来源复选框。
5. 点击确定运行分析。生产力提示在分析过程中显示。准备就绪后，[依赖查看器](#)将打开一个特殊的选项卡，使您可以检查依赖关系。
6. 在Dependency Viewer的左窗格中，选择要查找的节点。在右侧窗格中选择范围以查找所选节点的用法。搜索结果显示在选项卡的下部窗格中。

分析循环依赖关系

循环依赖关系分析使您能够检测指定范围内的包之间的任何循环关系。分析结果显示在[依赖查看器](#)的专用选项卡中。

分析项目的循环依赖性

1. 在主菜单上，选择分析 | 分析循环依赖。
2. 在“[指定循环相关性分析范围](#)”对话框中，选择所需的分析范围。
3. 点击确定运行分析。生产力提示在分析过程中显示。准备就绪后，[依赖查看器](#)将打开一个特殊的选项卡，使您能够检查依赖关系。
4. 在Dependency Viewer的左窗格中，选择要查找的节点。在右侧窗格中选择范围以查找所选节点的用法。搜索结果显示在选项卡的下方窗格中。

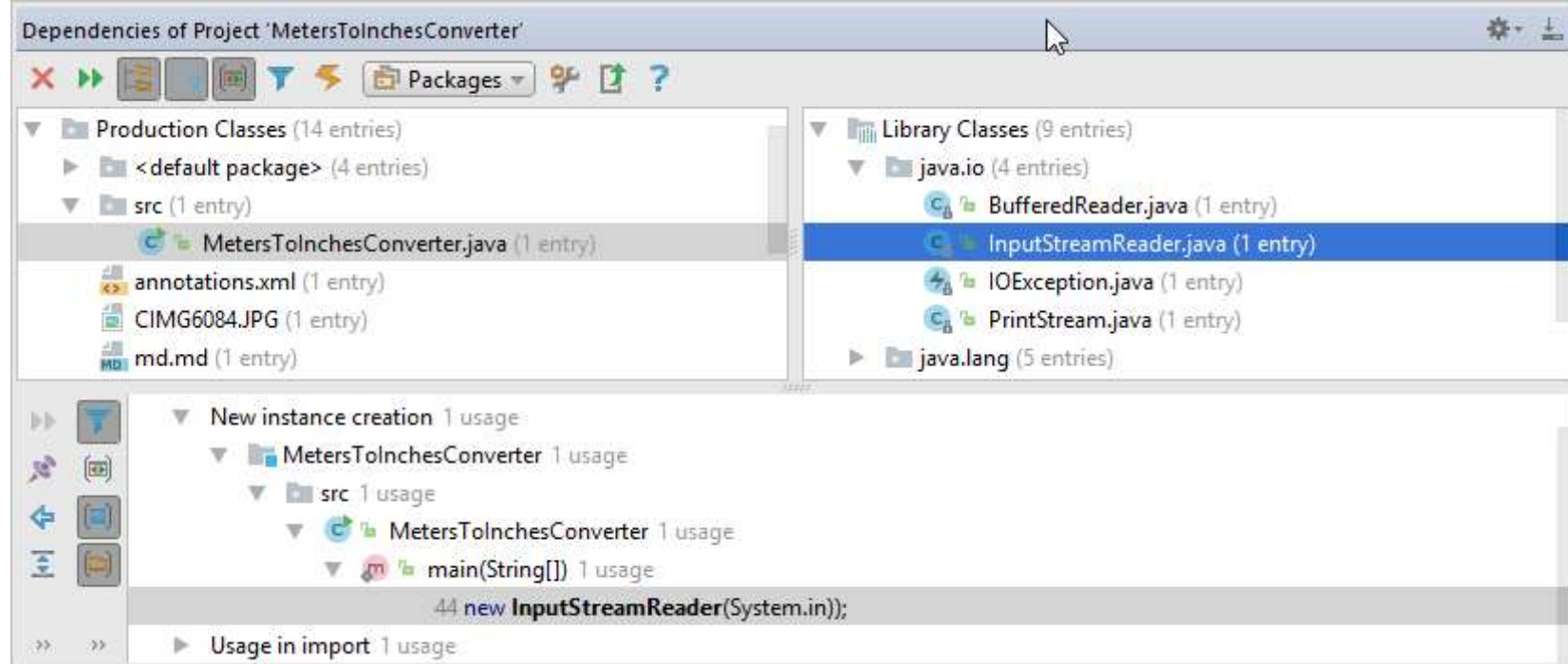
最后修改日期：2017年12月22日

分析依赖关系

IntelliJ IDEA使您能够分析项目的源代码并检测应用程序参与的依赖关系。每个依赖性分析的结果显示在[Dependencies Viewer](#)的一个单独的选项卡中。

分析项目中的依赖关系

1. 在主菜单上，选择分析 | 分析依赖关系。或者，右键单击要分析的元素（包，类等），然后在[项目工具窗口](#)的上下文菜单中或在编辑器中选择相同的命令。
2. 在“[指定相关性分析范围](#)”对话框中，选择所需的分析范围。
3. 检查依赖[关系查看器](#)中的[依赖关系](#)。



最后修改日期：2017年12月22日

分析重复

此功能仅在**Ultimate**版本中受支持。

在本页：

- [概观](#)
- [搜索重复](#)
- [在飞行中检测重复](#)

概观

IntelliJ IDEA帮助您在一定范围内找到重复的代码块。此范围可以是单个文件，项目，模块或自定义范围。分析结果显示在[Duplicates工具窗口](#)的专用选项卡中。

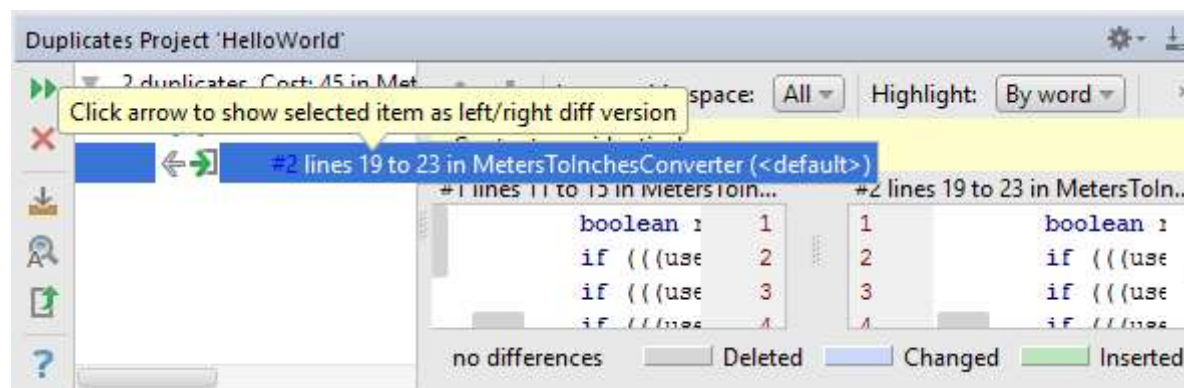
搜索重复


搜索重复

1. 执行以下任一操作：

- 在主菜单上，选择分析 | 找到重复。

- 选择分析 | 在编辑器的上下文菜单或[项目工具窗口](#)中找到“重复”命令。
2. 在“[指定代码复制分析范围](#)”对话框中，指定分析范围（整个项目，当前文件，未提交的文件（用于版本控制下的项目）或一些自定义范围）。另外，您也可以将测试资源纳入分析。
准备好后点击确定。
 3. 在“[代码复制分析设置](#)”对话框中，执行以下操作：
 1. 选择语言进行分析。
 2. 对于每种语言，请检查选项以定义分析的偏好。
例如，您可以选择请求代码片段的相同匹配被视为重复项，或者指定某个限制，低于该限制，代码构造不会被视为重复项（以避免报告**if**源代码中的每个构造）。点击OK。
 4. 在“[重复](#)”工具窗口中，浏览搜索结果。

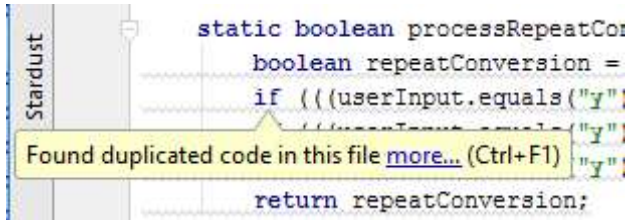


- 在工具窗口的左窗格中查看重复项列表。
- 查看右侧窗格中找到的重复项之间的差异。使用箭头按钮将选定副本放置在差异查看器的其中一个部分中，并比较代码的片段。
- 导航到编辑器中的重复项，使用跳转到源或 重复上下文菜单的显示源命令。
- 通过单击并在“提取方法”对话框中指定方法名称和参数，从源代码中消除重复项。这个过程类似于[Extract方法重构](#)，唯一的区别是在重复分析的情况下重复的代码块被自动找到。

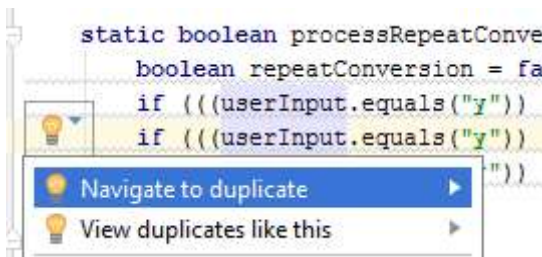
在飞行中检测重复

IntelliJ IDEA可以实时查看重复数据。这是通过[检查 General](#) | 完成的 重复的代码。

如果您偶然发现了一个副本，或者通过编写或粘贴代码创建一个副本，您将立即知道：

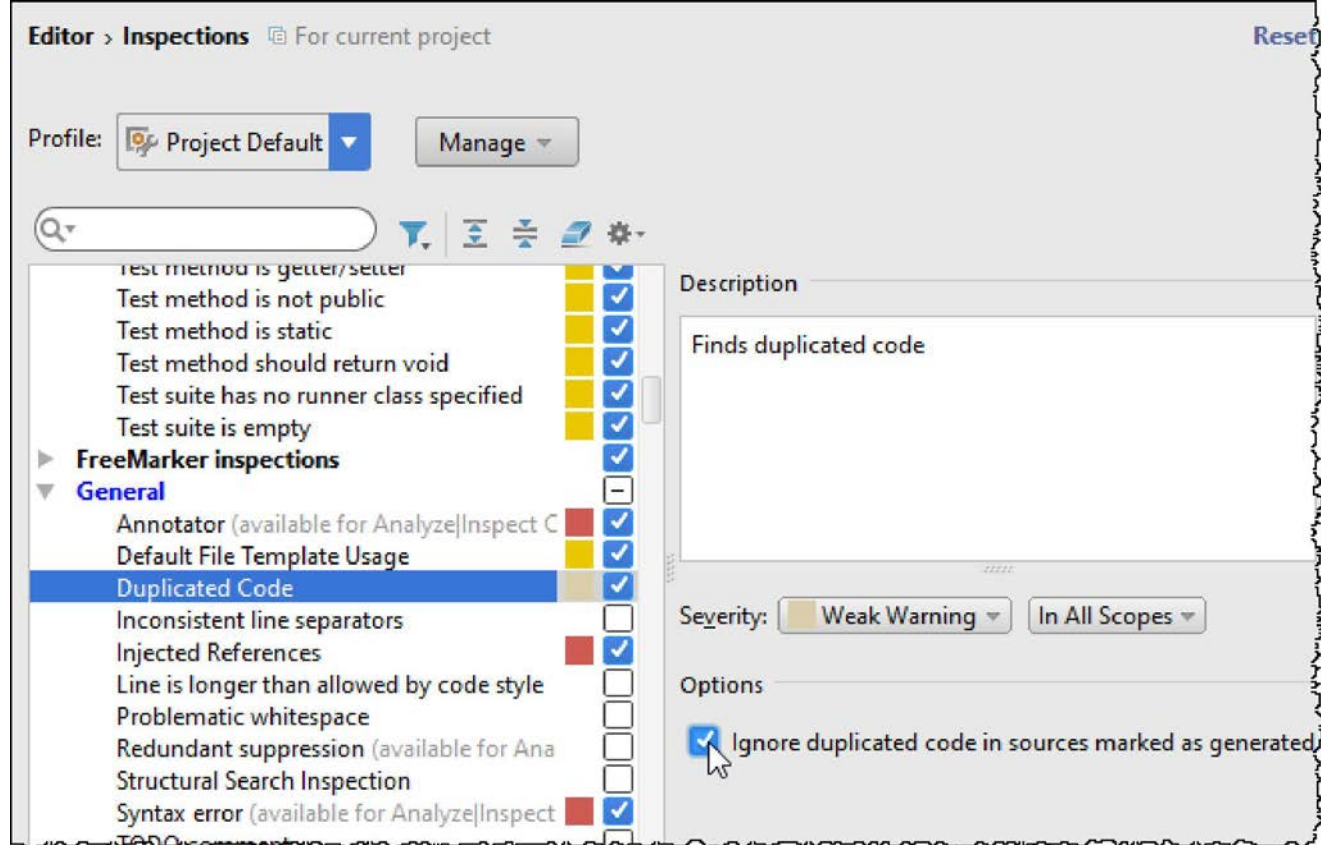


检查伴随着快速修复，使您能够导航到检测到的重复项，或在“[查找工具窗口中](#)查看所有重复项：



请注意，IntelliJ IDEA有助于避免在生成的源中找到重复项。

为此，请选中检查设置页面中标记为生成的源中的复选框忽略重复代码：



最后修改日期：2017年12月22日

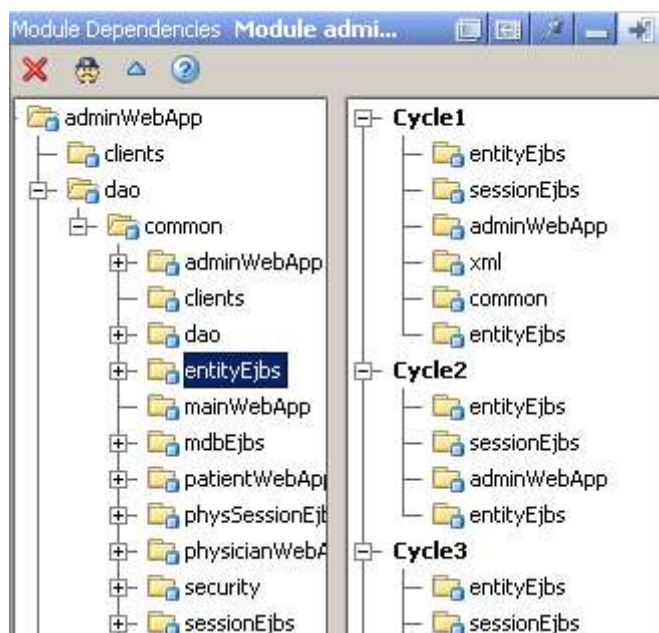
分析模块依赖关系


模块相关性分析显示指定范围内存在的所有模块，这些模块之间的关系（如在[“项目结构”对话框](#)的“[相关性](#)”选项卡中指定的）以及模块之间的循环依赖关系。

您可能想要使用这种类型的分析来确保您之前定义的依赖关系仍然存在于您的项目中。

分析模块依赖关系

1. 在主菜单上，选择分析 | 分析模块依赖关系。
2. 指定分析的范围。您可以选择整个项目或特定的模块。
3. 检查[模块相关性工具窗口](#)中的[依赖关系](#)



4. 在树视图选择一个模块，并使用“模块相关性”工具窗口的工具栏按钮来[查找依赖于所选模块的模块](#)。

最后修改日期：2017年12月22日

分析外部堆栈跟踪

在本页：

- [概观](#)
- [分析外部堆栈跟踪](#)

概观

您可能需要分析其他人（例如QA工程师）收到的异常，或调查死锁或挂起问题。与在调试模式下或运行单元测试时遇到的异常不同，这些异常没有链接可帮助您导航到源代码中的相应位置。而且，源代码可以被加密。

使用IntelliJ IDEA，您可以简单地复制异常或完整线程转储，将其粘贴到Stacktrace Analyzer，浏览信息并导航到相应的源代码。

分析外部堆栈跟踪

分析外部堆栈跟踪或线程转储

1. 在主菜单上，选择 分析 | 分析堆栈跟踪 ..

2. 在打开的**Analyze Stacktrace**对话框中，将外部堆栈跟踪或线程转储粘贴到放置堆栈跟踪或完整的线程转储 文本区域。
3. 指定是否要解扰堆栈跟踪。为此，选择 **Unscramble stacktrace**复选框，选择所需的解扰器和日志文件。
4. 如果在使用某些软件（例如，错误跟踪器或邮件客户端）进行处理之后，堆栈跟踪文本已损坏（行被剪切或打包，或者太长等），请单击“ 标准化”。
5. 点击**OK**。堆栈跟踪显示在“ [运行](#)”工具窗口中。

最后修改日期：2017年12月22日

在线程转储的情况下，IntelliJ IDEA将以一种可读的方式呈现所有线程，并对它们进行排序，以便首先显示那些最有可能造成问题的死锁或挂起问题。

分析数据流

在本页：

- [介绍](#)
- [分析数据流](#)
- [检查数据流分析的结果](#)
 - [数据流到这里](#)
 - [这里的数据流](#)

介绍

IntelliJ IDEA提供数据流分析功能来帮助您进行代码考察 - 更好地了解继承的项目代码，解释代码的复杂部分，找到源代码中的瓶颈等等。

具体来说，数据流往/从这里功能允许您：

- 查看分配给变量的值来自哪里。
- 找出变量可能具有的所有可能的值。
- 找出一个表达式\变量\方法参数可以流入的地方。

- 揭示潜在**NullPointerException**可能发生的地方。

如果要追溯传递给插入符号参数的值，可以使用**Analyze**分析工具创建源代码的 **切片**视图。数据流到这里命令。而且，使用 **Analyze |** 来自 **Here** 命令的数据流你可以找到一个表达式可以流入的位置。每个数据流分析的结果显示在**分析数据流**工具窗口的专用选项卡中。

分析数据流

要分析符号的数据流：

1. 打开所需文件进行编辑，请参阅 [编辑器基础知识](#)。
2. 将脱字符号放在要分析的符号（`expression \ variable \ method`参数）中。
3. 在主菜单或上下文菜单上，选择**分析 | 数据流到这里** 或 **分析 | 这里的数据流** 取决于你的目的。
4. 指定分析范围并选择是否要忽略来自测试代码的所有值。
5. 点击**OK**。在专用分析数据流工具窗口中[查看](#)分析结果。

检查数据流分析的结果

以下部分将简要介绍如何“读取”数据流分析结果。

- [介绍](#)
- [分析数据流](#)


- [检查数据流分析的结果](#)

数据流到这里

- 展开树挖掘导致符号的分配和方法调用链。具有灰色背景节点表示重复项（已经存在于另一个位置的树中的用法）。下图显示了 **Dataflow to Here** 分析结果的示例：



在以下的方向上的该视图流中的值：该字段的值 `String authType` 在 `SingleSignInMessage` 来自 `this.authType = authType` 赋值语句中 `SingleSignInMessage.setAuthType()` 的方法调用从 `ClusterSingleSignIn.register()` 方法用 `authType` 参数等。

- 要找出符号可能具有的值，请单击 **Dataflow** 工具窗口 主工具栏上的 **Group By Leaf Expression** 按钮。
- 要查看分配代码和方法调用，请按  切换按钮。IntelliJ IDEA 添加了一个预览窗格，其中显示了树中当前选定的分配或调用的代码，代码高亮显示：
- 要导航到任务或调用的源代码，请双击树中的相关行。

这里的数据流

分层视图与“数据流到这里”分析的结果类似，但是值相反。




最后修改日期：2017年12月22日

验证依赖关系

在本页：

验证依赖关系

1. 在[依赖查看器](#)的工具栏中，单击编辑规则  按钮。 [依赖项验证对话框](#) 打开。
2. 定义禁止和允许使用的范围：
 1. 单击 拒绝/允许部分中的添加按钮以添加默认范围。
 2. 单击 **Deny / Allow usages** 列中的范围条目，然后 从下拉列表中选择一个预定义的范围，或单击省略号按钮并在[Scopes](#) 对话框中定义范围 。
 3. 单击 **in / only** 列中的范围条目 ，然后选择一个预定义的范围，或者单击省略号按钮并在[Scopes](#) 对话框中定义范围 。
 4. 使用“ 添加 ” 和 “ 删除 ” 按钮组成完整的列表。
3. 应用更改。依赖分析重新执行新的规则。

最后修改日期：2017年12月22日