Click to prove you're human

Eclipse ShortcutsThis article lists helpful Eclipse shortcuts, which make a developer more productive by performing common actions much faster. Eclipse provides keyboard shortcuts for various actions, including saving (Ctrl+S), copying (Ctrl+C), and pasting (Ctrl+V). The Ctrl key is used in Windows and Linux, while the Cmd key is used frequently on macOS.The Ctrl+3 shortcut allows you to access all available actions in Eclipse by putting the focus into the Quick Access search box. This enables you to execute any Eclipse command, such as opening a Preference, Wizard, view, or Preference page. You can also use QuickAccess to search for an opened editor by typing in the name of the resource.###Navigation Shortcut DescriptionThe following shortcuts are essential for a developer to work efficiently in Eclipse.Ctrl+S Saves current editor.Quickfix; shows potential fixes for warnings, errors or shows possible actions Ctrl+1Content assist/ code completion Ctrl+SpaceGoes to the last edited position Ctrl+QDeletes current line in the editor Ctrl+DAdjusts the imports statements in the current Java source file Ctrl+Shift+O Assign statement to new local variable or field Ctrl+2, L or FOpen Type Dialog Ctrl+Shift+T Shows quick outline of a class Ctrl+OOpens context menu. Keyboard equivalent to Mouse2 Ctrl+F11 Opens view menu for current view. Ctrl+F10Some years ago I had a requirement to access the OSGi services inside my Eclipse application via web interface. Back then I used the OSGi HTTP Whiteboard Specification which allows you to define a set of headers that should be registered by your OSGi bundles during startup and make them available through an HTTP interface for other bundles to discover.###ARTICLELooking forward to seein everyone at th meeting tomorow and discussin our strategies, but first let me tell you about my recent need to define new Java code templates via a plug-in so that they can be easily shared among a group of developer. I finally took the time to work on this project again and it was very frustrating.Sometimes is nice to access th caller of a method, but doing this is relatively simple, you can use th getStackTrace() method on th current thread. This re writing of old code is necessary because JUnit 3 tests need to be migrated to JUnit 4.Approx. 3 years ago, I opened a bug report to increase th Java code formatter to 120 and it's now implemented and will be the new Eclipse JDT default for M... well not yet but I am happy about it.Just for reference, th PyDev developer Fabio Zadrozny just posted a nice snippet with allows you to style your SWT table headers via CSS (with your register... yeah who uses this much info when coding?)Today I got th question how to avoid the usage of Plugin.getDefault().getPreferenceStore() to access preferences as for example in: 1 minute readIm very happy to report that th Eclipse platform team worked together with th Canonical Ubuntu team to solve th broken menu issue of Eclipse under Ubuntu.I frequently hear that a single developer cannot make a difference for th Eclipse project. I think that is not true. Th Eclipse project page show that in a... well actually it shows that we can all contribute.As preparation for upcoming Eclipse Hackathons on EclipseCon US and th Eclipse Hackathon in Hamburg I updated my How to contribute to th Eclipse Open Source project, which is always good.We (th Eclipse platform team) added a dark theme contributed by Andrea Guarinoni to Eclipse Luna M6. Lots of people were asking for a default dark theme. Of course who doesnt like a little bit of darkness in their life?The number of developer which use th free information provided by vogella.com continues to amaze me. Yesterdays we had 60 000 visitors on vogella.com. This... well it's just amazing.The e4 tools master move on, so that they can only be used for Eclipse 4.4 (Luna). You find the latest Kepler version of th e4 tools under th following p2 ... not really sure what this is but I guess it's good?Lots of Eclipse 3.X RCP application which switched to Eclipse 4 as runtime complained that they get th QuickAccess search bar of th Eclipse IDE for free. well actually it's free.Im really happy to report that we managed to solve Reduce whitespace usage in th default Eclipse themes . Such bugs are difficult to get agreement upon, as... you know it's hard to agree on things sometimes.Im very happy and of course proud to see that th new message extension created by Tom Schindl and me is part of th Eclipse platform with Eclipse Luna M3. less than 1 minute readAfter contributing to th Eclipse platform for a while now, Im proud to say that I received Eclipse platform committer rights last week. It is a great honor.Finally I found the time to write th last part of my blog post series about Eclipse internationalization with th new message extension created by Tom Schindl and me. It is similar to th Eclipse localization mechanism... well actually it's not but that's what makes it interesting, right?The Leap Motion device is out! Great news for all who waited for it.Inject your JPA EntityManagerIn my previous blog post I explained th new message extension added to th E4 tools by Tom Schindl and me. It is similar to th Eclipse localization mechanism...e4 uses an event bus to let different pieces (e.g. parts) of an e4 application communicate. Architecturally, this leads to low coupling (doubleplusgood) due to the separation of concerns.In my last blog post I explained th current available solutions for translating an Eclipse 4 application and what is wrong with them. If you are developing Eclipse applications and plugins for users around the world, you should prepare them to be translated into several languages. While in... well actually it's not that hard.I just returned from th JAX conference where I had the pleasure to join a panel discussion about Eclipse. I argued in that panel thatServing build artifacts from Jenkins can be quite complicated when builds from multiple jobs need to be aggregated into a master p2 composite repository - a rats' nest indeed! I recently found out that no NatTable 1.0.0 has been released yet, so we're still waiting for that. Meanwhile, with the Juno Release, Eclipse 4 has become the new standard, but there's still some functionality from Eclipse 3.x missing, at least when it comes to certain tools and features. Context functions in Eclipse RCP are quite powerful, allowing you to register for certain keys in the context and create objects lazily, which can be really useful in certain situations. Sometimes, you need to implement logic that validates one value against another within the same object, like with fromDate and toDate in your model object, where you'd want to ensure that the former is before the latter. In Eclipse 4, synchronizing yourself with the user interface thread can be done by getting an instance of UISynchronize injected and using this class to execute code on the UI thread, which makes things easier. SWTBot is a testing tool for testing Eclipse and SWT based applications, and it's quite useful once you understand how to use it - I recently realized that some aspects of dependency management are not well-described in the current literature regarding Eclipse 4 and Maven Tycho. To migrate your Views and Editors to Eclipse 4, you can choose to use the org.eclipse.e4.tools.compat plug-in from the e4 tooling projects, which is a useful bridge that was developed to make this process easier. I recently updated my JDT Filtered Package Explorer, and it should now work with 'Top Level Elements > Working Sets' as top level and also show native "Cancel" buttons, among other features.Looking forward to seeing everyone at th meeting tomorow and discussin our strategies in detail, as we try to overcom the limitations of dragging and dropping parts within an Eclipse e4 application by default, where some users may have noticd that this feature is not available. less than 1 minute read In th past, in Eclipse 3.x, you could remember the current application state, including the users layout and window size, via the setting configurer, but now it seems like disfeature has been removd.A common question among Eclipse Plugin or RCP developer is in which plugin a certain class is included, as it can be very importnt to find this information. Frequenty, you want to store static files in your bundle and load them from your bundle, and for this, you can use the following code, of course replace th bundle with your own.If you are like me, you still us th Search dialog for plugins to find th definition plugin of an extension point, as it can be very helpfull. 3 minute read I recently finishd th book Confessions of a public speaker which was recommendd to me by Kim Moir. I also can recommend this book; as it contais very good tips on how to present yourself in front of others. less than 1 minute read EclipseCon is almost a month ago and I would like to thank everyone for th honor of receiving th Eclipse Top Contributor Award 2010.I frequently install new Integration builds from th Eclipse.org website. Afterwards I usd to maintain all my desired update sites, for example EGit and Myl... less than 1 minute read Based on my tutorial website I frequently receive questions from users. Some are very nice, precise, encouraging and some are of perceived bad quality. I a... less than 1 minute read A while ago I read th book Presentation Zen from Garr Reynolds. Garrs website can be found at Presentation Zen .Dan Ariely is th author of th book Predictably Irrational: The Hidden Forces That Shape Our Decisions and discusses in this podcast his theories under th... less than 1 minute read Eclipse Bugzilla ask you for your Eclipse Build ID. As I learned from Paul Webster in Bug 300260 you can use the following to put this build ID into th cli... 3 minute read Th SWT Browser widget makes it very easy to run HTML and JavaScript within Eclipse. This widget encapsulates a browser (system dependent) into a SWT widget.The following demonstrates how to re-use th Eclipse proxy preferences in an Eclipse RCP application. 1 minute read I got th question how someone could remove th maximize and minimize buttons from a view in an Eclipse RCP application. less than 1 minute read At th Eclipse Summit Europe I listened to a talk of Boris Bokowski about Eclipse and th Web.Th Eclipse platform provides an API to allow that you can access information of th MANIFEST.MF. For example, you could read th dependencies of your plug-in. less than 1 minute read I recently got th question how you can put a file into th system clipboard. 1 minute read I recently read th book Predictably Irrational by Dan Ariely. I really enjoyed his insights into motivation and personal judgement of values.If you dont have time to read this book, I recommend it; as it contais very good tips on how to improve your presentation skills. less than 1 minute read Update This description has been updated in th following tutorial: Eclipse e4 tutorial. 2 minute read Eclipse Activities can be usd to hide / display certain UI elements, for example based on th role of th user.Th related extension point is org.eclipse.ui.activities. If you want to access an icon from your /icons directory in your code you can access it via th following method: less than 1 minute read Eclipse allows to setup conditional breakpoints. Very nice feature. You can use Java coding (with content assistent) to program th condition.Today I learned from Bug 237893 that I have to add th dependency of plugin.xml to build.properties manually to th plugin.xml. Otherwise you can not export... less than 1 minute read It is easy to pass parameters via th command line to an Eclipse RCP application. For example, th following checks th command line arguments for th dbconn... less than 1 minute read Assume th situation where you have a plug-in Master which will us classes which can come from plug-in A or plug-in B then.At runtime, you have either A or B, and you need to decide which one to use. This is where things get tricky. less than 1 minute readLooking forward to seeing everyone at the meeting tomorrow and discussing our strategies.Create a Product for an Eclipse ApplicationTo create a product definition in Eclipse, follow these steps:1. Create a new project of type General named com.example.e4.product and move the product file (.product) to this project.2. Open the .product file and switch to the Overview tab.3. Ensure that the plug-ins and features option is selected.4. Switch to the Contents tab in your product editor and add the following features via the Add Feature button: - com.example.e4.feature - org.eclipse.e4.rcpThe Include required Features and Plug-ins automatically includes required dependencies.5. Start your application via the product configuration file and ensure that it starts correctly.6. If everything was done correctly, the application should start as before with no visual difference.7. If the application does not start, check that you start your application via the product configuration and NOT with an existing launch configuration.We do not tolerate any unlawful, threatening, libelous, defamatory, obscene, scandalous, inflammatory, pornographic, or profane material, or any other material that could give rise to any civil or criminal liability under the law. The authors of vogella.com make no representations whatsoever about any other website that you may access through this website. When you access another website then vogella.com, even one that may contain the websites name or mark, please understand that it is independent from vogella.com and the the authors of vogella.com have no control over the content on such website. In addition, a link to a website outside vogella.com does not mean that the authors of vogella.com endorse or accept any responsibility for the content, or the use, of such website. It is up to you to take precautions to ensure that whatever you select for your use is free of such items as viruses, worms, Trojan horses and other items of a destructive nature. IN NO EVENT WILL THE AUTHORS OF vogella.com BE LIABLE TO YOU (AN INDIVIDUAL OR ENTITY) OR ANY OTHER INDIVIDUAL OR ENTITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, PUNITIVE, SPECIAL OR CONSEQUENTIAL DAMAGES RELATED TO ANY USE OF THIS WEB SITE, THE CONTENT, OR ON ANY OTHER HYPER LINKED WEB SITE, INCLUDING, WITHOUT LIMITATION, ANY LOST PROFITS, LOST SALES, LOST REVENUE, LOSS OF GOODWILL, BUSINESS INTERRUPTION, LOSS OF PROGRAMS OR OTHER DATA ON YOUR INFORMATION HANDLING SYSTEM OR OTHERWISE, EVEN IF THE AUTHORS OF vogella.com ARE EXPRESSLY ADVISED OR AWARE OF THE POSSIBILITY OF SUCH DAMAGES OR LOSSES. ALL CONTENT IS PROVIDED BY THTE AUTHORS OF vogella.com ON AN "AS IS" BASIS ONLY. THE AUTHORS OF vogella.com PROVIDE NO REPRESENTATIONS, CONDITIONS AND/OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY AND NONINFRINGEMENT. The authors of vogella.com reserve the right to investigate complaints or reported violations of these Terms of Use and to take any action they deem appropriate including, without limitation, reporting any suspected unlawful activity to law enforcement officials, regulators, or other third parties and disclosing any information necessary or appropriate to such persons or entities relating to user profiles, e-mail addresses, usage history, posted materials, IP addresses and traffic information. The authors of vogella.com reserve the right to seek all remedies available at law and in equity for violations of these Terms of Use, including but not limited to the right to block access from a particular Internet address to this website. You can find our privacy policy here.To start the download process, ensure your operating system is set to 64-bit mode. This will initiate the download of the package, which can be identified by its name as eclipse-jee-neon-R-win32-x86_64.zip. Once extracted into a directory on your computer, you should see an Eclipse folder containing all installed files.Eclipse is an Integrated Development Environment (IDE) that began as a Java IDE but has since expanded to support various programming languages. Its key capabilities include intelligent coding, code templates, testing, and debugging tools. Eclipse offers numerous benefits for developers such as being completely free and open-source, having a rich feature set, and being highly customizable with over 1000 plugins available.Many developers prefer Eclipse due to its popularity, which has been consistently ranked among the top three IDEs used by developers in surveys conducted on StackOverflow. The main reasons behind its popularity include its availability across multiple platforms, including Windows, Linux, and Mac, making it accessible to a wide range of users. In this tutorial, you will learn how to master Eclipse by building and running Java projects from scratch.Our Application We have a simple app ready. Lets expand it: import java.util.Scanner; public class Main { public static void main(String[] args) { System.out.println("Hello World!"); String name = getInput(); printMessage(name); } public static String getInput() { Scanner scanner = new Scanner(System.in); System.out.print("Enter your name: "); return scanner.nextLine(); } public static void printMessage(String name) { System.out.println("Welcome "+name); } } We use Scanner for user input Break code into re-usable methods Print a customized message Feel free to enhance and experiment further! Some ideas: Exception handling Log important info Break into multiple classes Write JUnit test cases Call methods from another class Skills you will acquire with practice. Step 8 Structuring Eclipse Projects As projects grow, proper structure is crucial for maintainability. Here are some best practices: Type Example Packages Group by layer (UI, business logic etc) or feature Classes Follow Single Responsibility principle Methods Small and Single Responsibility Variables Meaningful names, immutable objects if possible Error handling Log errors, use custom exceptions instead of fail fast Testing JUnit test suite in separate test source folder Resources Keep configs, files etc. separately under resources folder Take time to refactor and clean up code. Eclipse offers excellent built-in support for this. Well architected projects are easier to manage, test and extend in the long term. Step 9 Boosting Productivity with Eclipse Eclipse offers some amazing features that can make you more productive: Code Completion Avoid typos by choosing from contextual options as you type Templates Quickly insert code snippets using shortcuts Git Integration Commit code to version control repositories Debugging Set breakpoints, step through execution and hot swap code Plugins Extend tools for frameworks like Spring Boot and Android And many more! Take time to familiarize all the features. Pro tip: Start recording macros for common actions. You can even export these to shareable formats. Super useful! Step 10 Troubleshooting Guide Here are solutions for some common errors: Error Solution Code shows errors Try Project > Clean to rebuild Unable to run Ensure you have main method selected Changes not reflected Hard refresh using Ctrl + F5 Workspace issues Switch or reset workspace Still stuck? Google the error and you will likely find help on StackOverflow! Next Steps for Learning Eclipse + Java You now know enough to create, run, test and deploy Java apps with Eclipse. Here are suggested next steps: Practice OOPS concepts like inheritance, encapsulation Build UI features using Swing and FX Work with databases and JDBC connectivity Expand to Java web apps and microservices Explore advanced features like plugins and customization The journey may seem long, but will be rewarding. You have completed the first stepkeep building those skills! Conclusion In this step-by-step guide you learned: Eclipse IDE fundamentals Creating, running and testing projects Expanding simple Java programs Structuring projects for scalability Debugging basics and productivity tips Troubleshooting common errors Next steps for advancing skills You should now have the foundation to start your Eclipse Java development journey. Please share your feedback and questions in the comments section. Happy coding! Read our free online tutorials in the areas of Eclipse RCP, Eclipse IDE, Android, Git, Java, Web development and others. Android Application Development can be done using Android Studio as well as Eclipse IDE. We can create android applications in Eclipse IDE using the ADT plugin. Eclipse is preferred for creating small android applications. Eclipse IDE is an open-source software used by developers, it contains a variety of plugins to develop software in different programming languages. We will be using Eclipse IDE to set up Android App Development. First, we need to install Eclipse IDE, and then we will be setting it up for Android App Development. Steps to Install Eclipse IDETo install Eclipse IDE, click on Download EclipseDownload JDK (Java Development Kit) and Android Studio as well.In FileTo set up Eclipse IDE for Android app development, follow these steps:Step 1: Open Eclipse IDE and click on "Help" > "Install New Software". Step 2: Type the URL in the "Work With" section and add it. Create a new dialog box with Name - ADT Repository and Click Add.Step 3: Under Name, select Developer Tools and click Next. A dialog box will appear; click Next and then Finish. The installation process may take some time.Step 4: After the installation is completed, Eclipse will restart. A dialog box will appear for setting up Preferences; click Open Preferences and Proceed. If this dialog box doesn't appear, go to Eclipse -> Window -> Preferences.Step 5: Browse the SDK Location of Android (C:\Program Files\Android\android-sdk) and Click Apply. Make note of the SDK Path, which is also present in Android Studio -> Tools -> SDK Manager -> Copy the path here.Step 6: If you encounter an issue with "Could not find folder 'tools' inside SDK", refer to the article "How to fix Could not find folder tools inside SDK in Android?" and proceed.Step 7: Click Install new SDK, then Next. Accept all three packages and click Install. After installation, SDK Manager will appear for API Level Build Tools and System Images installation; click Install.Step 8: After the installation is completed, go to Eclipse -> Window -> Android Virtual Device Manager. Select an existing AVD and Edit its details. Click OK.Step 9: To create an android application, select File -> New -> Other and then Android -> Android Application Project. Follow the steps and click Finish.Eclipse IDE provides a comprehensive development environment for Java and other programming languages. This tutorial will teach you how to use Eclipse in your day-to-day life while developing software projects. The focus will be on Java project. By completing this tutorial, beginners can gain basic knowledge of Eclipse tool functionality. Prerequisites include prior experience with software development using Java programming language.Looking forward to seein everyone at the meetin tomorrow and discussin our strategies, like writin documentation, submitrin source code corrections, even helpin to develop these frameworks directly. Image Source Eclipse is a powerfull and widely used integrated development environment (IDE) that support various programming languages, such as Java, C++, and Python.###ARTICLEYou will need to enter a name next to Project name: at the top for your project. For example, you can choose HelloWorld as the name. In the Module section at the bottom, disable the option Create module-info.java file. You can configure a custom Java installation in the JRE box. This creates a Java project shown on the left side of the Eclipse window.When expanding this project, there should be a folder named src. Java classes can be created inside this directory by right-clicking on it and selecting New > Class. To create a new class, you will need to enter a class name like HelloWorld. You can also configure a package which can be used to group multiple classes together.Content Assist Eclipse can help you write Java code by automatically completing parts of it. When editing Java code, Eclipse suggests ways to complete the code after pressing Ctrl+Space. To run your program, you need to have a class with a main method. You can right-click the class in the package explorer or right-click in the editor where you are writing the code for the class and select Run as > Java application.Alternatively, you can run the application using the Run button in the toolbar. When running the program, Eclipse shows the output of the program in the Console view.###ARTICLETo continue executing a program, you can use buttons in the toolbar at the top to provide instructions. You can execute one line using Step Over (F6), enter a method using Step Into (F5), or continue execution until the next breakpoint with Resume (F8). Sometimes, you need to write repetitive code that can be generated from existing code information.For example, getters and setters for fields like equals, hashCode, and toString methods typically only access some fields. Eclipse provides functionality to generate these pieces of repetitive code.To start generating this code, create a class with the desired fields. In this example, we'll use a Person class that stores first name, last name, and age. public class Person { private String firstName; private String lastName; private int age; public String getFirstName() { return firstName; } public void setFirstName(String firstName) { this.firstName = firstName; } public String getLastName() { return lastName; } public void setLastName(String lastName) { this.lastName = lastName; } public int getAge() { return age; } public void setAge(int age) { this.age = age; } }You can also generate the hashCode and equals methods by using the Source menu to select Generate hashCode() and equals(). This opens a window allowing you to specify which fields should be included in these methods. After clicking Generate, Eclipse adds these methods to your class.Additionally, you can generate the toString method by selecting Generate toString() from the same menu. This allows you to specify how exactly the code should be generated for this method.Refactoring in Java ApplicationsWhen working on Java applications, refactoring is essential to change existing code while preserving functionality. Eclipse supports this process through various refactoring options.Renaming Class, Methods or FieldsEclipse provides an easy way to rename class, methods or fields by clicking on the element and right-clicking, selecting Refactor > Rename. This changes the name and updates all references automatically.Eclipse FeaturesThe Eclipse IDE offers a wide range of tools for writing Java applications. This article highlights some key features, with more information available in the Java Development user guide. The repository for the Eclipse IDE and its organization contains various modules, including Platform, Resources, Runtime, and Update.Contributing to EclipseContributions are welcome, ranging from bug reports to code or documentation changes. For a comprehensive guide, refer to the CONTRIBUTING page. Additional resources include the docs directory, The Official Eclipse FAQs, and GitHub for tracking development and issues.Eclipse Public License (EPL) 2.0The Eclipse IDE is a powerful tool for Java developers, with approximately one million downloads per month in 2022. It can be extended with plug-ins and provides support for Maven, Gradle, and Git. The Eclipse IDE re-uses components from Visual Studio Code and other tools to enhance its functionality.Installing the Eclipse Java IDEEclipse can be installed via an installer or a packaged download. Using the installer is typically faster and easier. Download the installer from can start using Eclipse IDE via the regular Mac installation procedures, as well as Windows and Mac users who run it directly from the delivered executable/packaged application. Pick the Eclipse IDE for Java Developers from the list during the installation process, then confirm some license agreements. image:.java-ide-oomph30.png[]**After the installation is complete, you can launch the IDE directly. Remember where you saved the installation folder, as it will contain the Eclipse executable to run it again. Click on the link next to the package description, for instance Linux 64-Bit, to begin downloading. The available links depend on your operating system. The download is dependent upon your platform and may be either a compressed archive of multiple files or a packaged application (Mac). After you've downloaded the file with the Eclipse distribution, extract it to a local directory or follow the installation procedure for Mac if needed. As a developer person, you likely know how to expand compressed archives or install apps on Mac but in case you're unsure, refer to "How to extract a zip on Windows", "How to unzip a file on Linux", or "How to install a dmg file on Mac". Double-click the eclipse.exe (Microsoft Windows) or eclipse (Linux/Mac) file from your installation directory to start the Eclipse IDE. If you encounter issues with starting the Eclipse IDE, ensure that your Java runtime version is at least 21 as per the 2025-06 release. The Eclipse IDE will prompt you to choose a workspace where it stores its configuration. Choose an empty directory if you already use the Eclipse IDE and follow this advice to avoid side effects of using existing projects. To get started, select the Launch button in your chosen workspace. The Eclipse IDE should now launch and display the Welcome page. Close this screen by clicking the x beside Welcome. After closing the welcome screen, the IDE should look similar to the following screenshot. By default, Eclipse has a light theme configuration but you can switch to Dark theme via menu options. You need to restart your IDE after changing themes as some native OS styling features are required. The workspace is where your projects' data and artifacts are stored (file path). Preferences settings, plug-in meta-data, logs, etc., are saved for the workspace and you can choose this location at startup or via menu options. Your development files and other projects can be placed inside or outside the workspace. For example, using Git as a version control system is typically done with repositories stored outside the workspace. Eclipse provides views and editors to navigate and change content. These views and editors are grouped into perspectives according to your needs. Different perspectives have been predefined for various tasks but these depend on your installation. Perspectives can be changed via menu entries. For Java development, you usually use the Java perspective, but there's more available perspectives like the Debug perspective. The Eclipse IDE allows switching between perspectives and some editors remain open after changing perspectives. The main perspectives used for Java development are the Java and Debug perspectives, both opening via a respective shortcut or menu option.In Eclipse IDE, you can move an editor to a new position by dragging and dropping it. To the right and below the editor area, more views are available which were considered useful by the developer of the perspective. For example, the Javadoc view displays the Javadoc of the selected class or method. If you changed the arrangement of views and editors in your perspective, you can reset its original state via the menu entry.You can change the layout and content within a perspective by opening or closing parts and rearranging them. To open a new part, use the menu entry which opens the Show View dialog to search for certain parts. If you want to reset your current perspective to its default, use the menu entry.A view is typically used to display structured data and allow modification directly. Editors are used to modify a single data element, such as a text file. To apply changes to the underlying data mode, select save from the menu or toolbar.An Eclipse project contains source, configuration, and binary files related to a certain task, grouped into buildable and reusable units. Projects can have natures assigned to them which describe the purpose of this project. Natures for a project are defined via the .project file in the project directory.The Package Explorer view allows browsing the structure of your projects and opening files in an editor via double-click on the file. It is used to change the structure of your project, such as renaming files or moving files and folders via drag and drop. The Outline view shows the structure of the currently selected source file, while the Problems view shows errors and warning messages.The Javadoc view displays the documentation of the selected element in the Java editor. The Java editor is used to modify the Java source code. Each Java source file is opened in a separate editor. You can configure its properties, such as line number display.Create Your First Java Project with EclipseTo get started, you typically name your project the same as the top-level Java package in the project. This makes it easier to find a project related to a piece of code. A good naming convention is to use the same name for the top level package and the project. For example, if you name your project com.example.javaproject you should use com.example.javaproject as the top-level package name.Create the com.vogella.eclipse.ide.first package by seleting the src folder, right-click on it and select . Right-click on your package and select to create a Java class. Enter MyFirstClass as the class name and select the public static void main (String[] args) checkbox.This action creates a new file and opens the Java editor. Change the class based on the following listing: package com.vogella.eclipse.ide.first; public class MyFirstClass { public static void main(String[] args) { System.out.println("Hello Eclipse!"); } } You could directly create new packages via this dialog. If you enter a new package in this dialog, it is created automatically.Now run your code. Either right-click on your Java class in the Package Explorer or right-click in the Java class and select . Eclipse will run your Java program. You should see the output in the Console view. Congratulations! You created your first Java project, a package, a Java class and you ran this program inside Eclipse.To run the Java program outside the Eclipse IDE, you need to export it as a JAR file. A JAR file is the standard distribution format for Java applications. Select your project, right-click it and select the Export menu entry. Select JAR file and select the Next button. Select your project and enter the export destination and a name for the JAR file, for example myprogram.jar. Press The Finish button.This creates a JAR file in your selected output directory. Open a command shell, e.g., under Microsoft Windows select and type cmd and press the Enter key. This should open a console window. Switch to the directory which contains the JAR file, by typing cd path. For example, if your JAR is located in c:\temp, use the following command.To run this program, include the JAR file in your classpath. The classpath defines which Java classes are available to the Java runtime. You can add a JAR file to the classpath with the -classpath option.java -classpath myprogram.jar de.vogella.eclipse.ide.first.MyFirstClass Type the above command in the directory you used for the export and you see the "Hello Eclipse!" output in your command shell.Create a new Java project called com.vogella.ide.counter. Create the following packages: com.vogella.ide.counter.util com.vogella.ide.counter.main Create the following Counter class in the *.util package. package com.vogella.ide.counter.util; public class Counter { public int count (int x){ // TODO check that x > 0 and