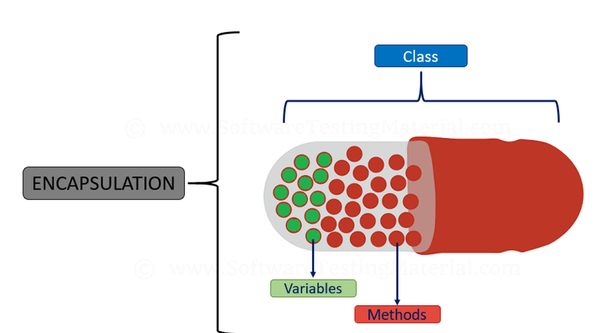
**Selenium API Wrapper**



***DAO MINH DAM  
Email:***[*daominhdam@gmail.com*](mailto:daominhdam@gmail.com)

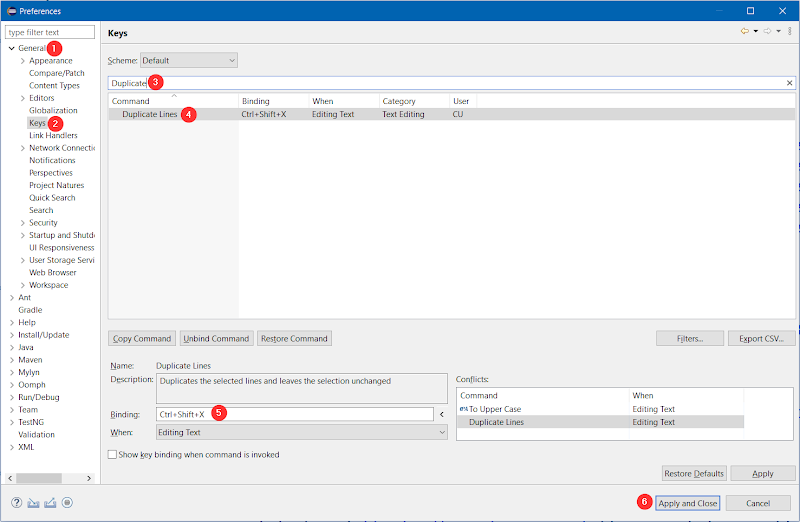
***Skype:****daominhdam (Đào Minh Đảm)*

[***https://automationfc.vn***](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://automationfc.vn%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193827000%26amp;usg%3DAOvVaw1vnszV6CDIpgsC1XjNX-9X&sa=D&source=editors&ust=1628669193888810&usg=AOvVaw08ArzJ0xNj4ecz0eBWnbfk)*-*[***https://automationfc.com***](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://automationfc.com%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193828000%26amp;usg%3DAOvVaw3ov324FIFPL9dxOkr1G-sd&sa=D&source=editors&ust=1628669193888914&usg=AOvVaw0FPbZnTh8a7fyUawx_KOJn)

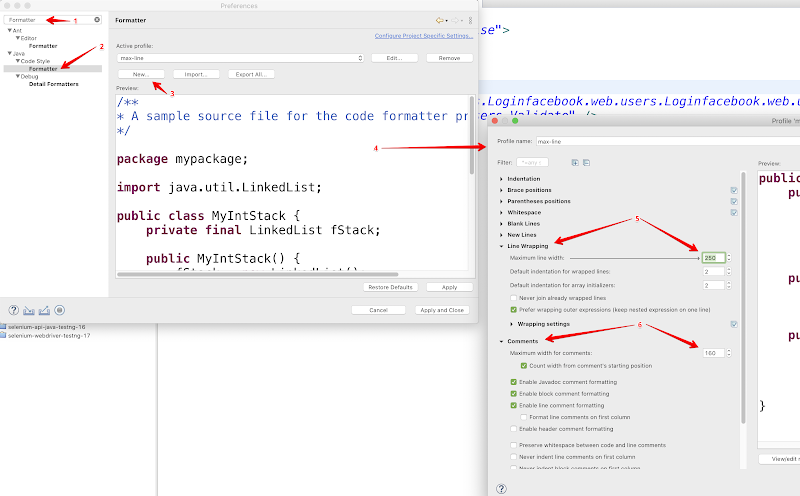
***Tài liệu này chỉ sử dụng trong khóa học - vui lòng không chia sẻ công khai***  
-------------------------------------------------------------------------------------------------------------------------------

### Eclipse Tips

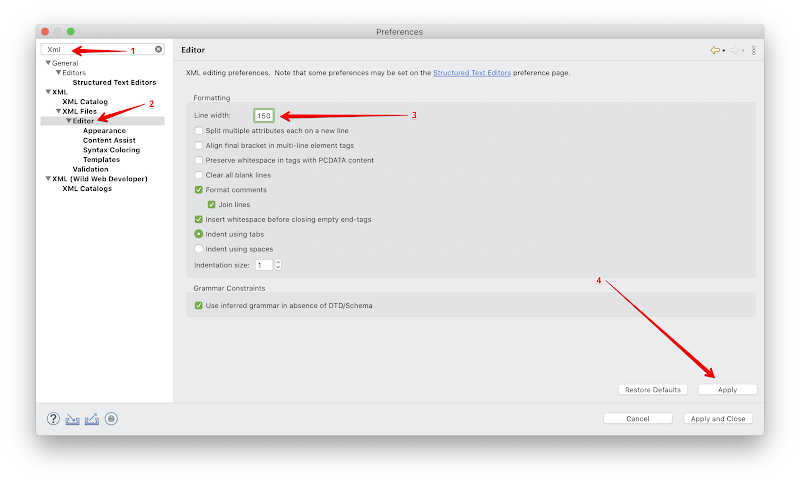
* **Builtin keyword**
* Ctrl + D: Xóa dòng/ khối lệnh
* Ctrl + Shift + F: Format dòng/ khối lệnh
* Ctrl + Shift + O: Import/ remove thư viện
* F3: Đi vào trong hàm/ biến
* Ctrl + Shift + C
* Comment/ Uncomment dòng/ khối lệnh
* //
* Ctrl + Shift + /
* Comment
* \*\* = Java Docs
* Ctrl + Shift + \
* Uncomment
* Ctrl + Space: Suggest code
* Alt + ↑ hoặc ↓: Move 1 dòng/ khối lệnh lên hoặc xuống
* Ctrl + (phím - hoặc phím +): giảm hoặc tăng font chữ
* **User Define keyword: Windows > Preferences > General > Keys**



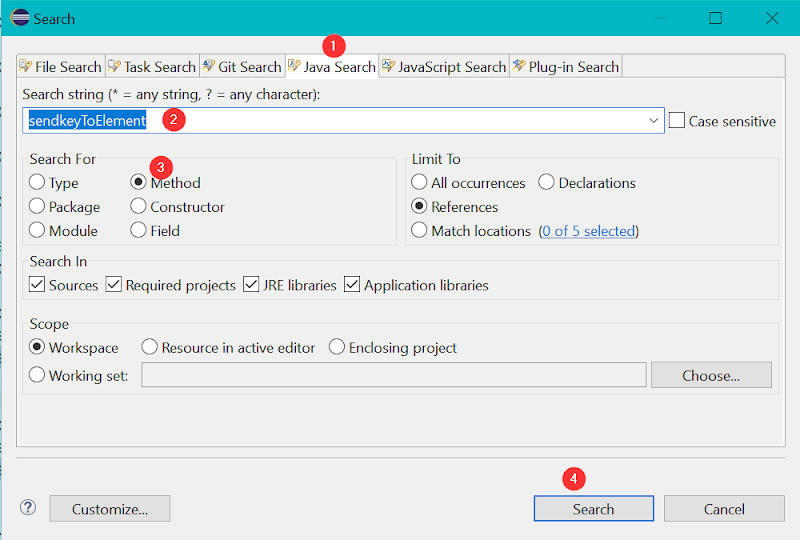
* Ctrl + Shift + X: Duplicate 1 dòng/ khối lệnh
* Ctrl + Shift + A: Collapse các method lại cho gọn gàng
* Alt + Shift + A: Expand các method lại cho gọn gàng
* **Tăng max-line**
* **Java**

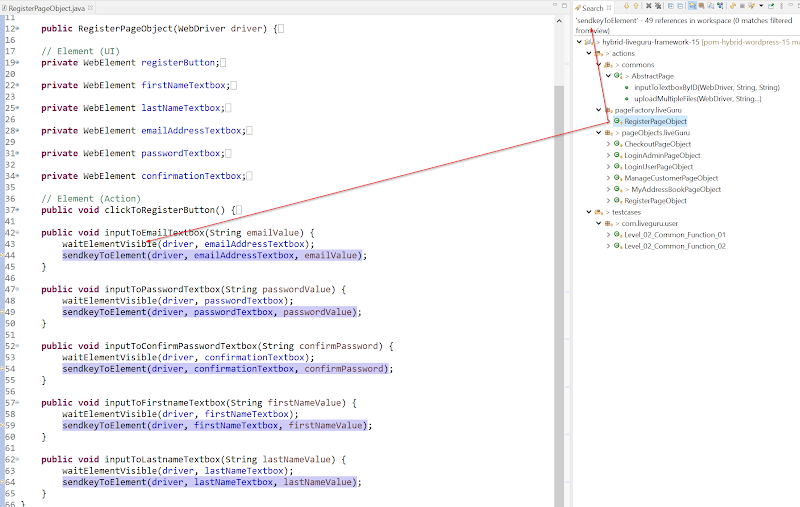


* **Xml**

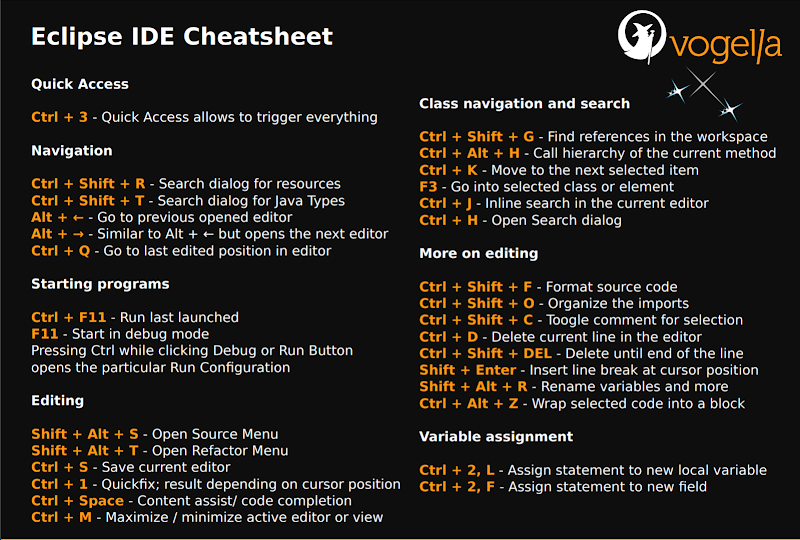


* **Search keyword/ method/ variable**
* **Ctrl - H**



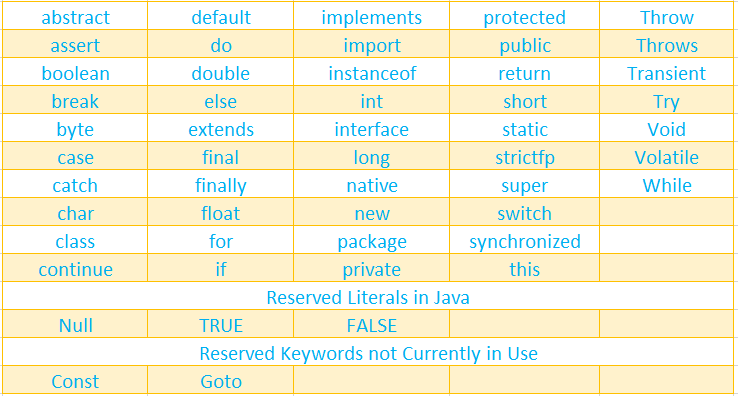


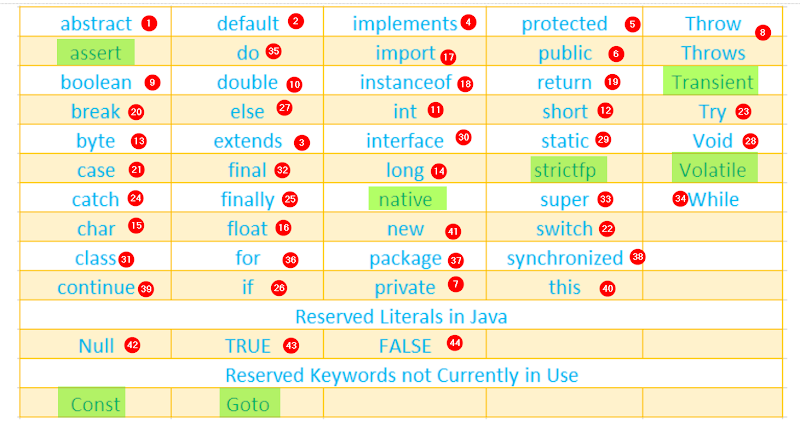
* **Cheatsheet**



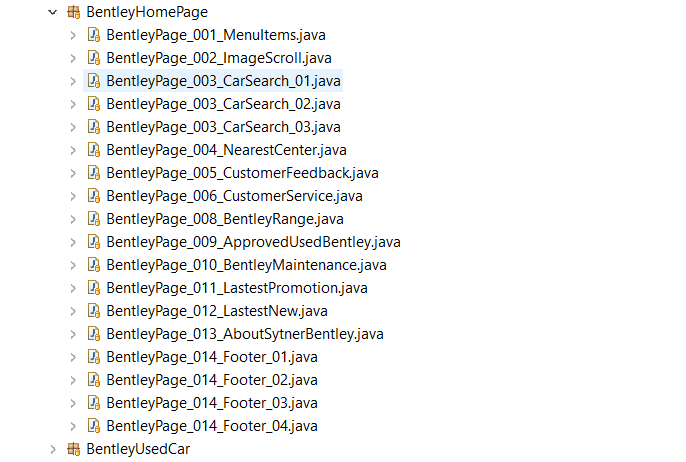
**Java Convention**

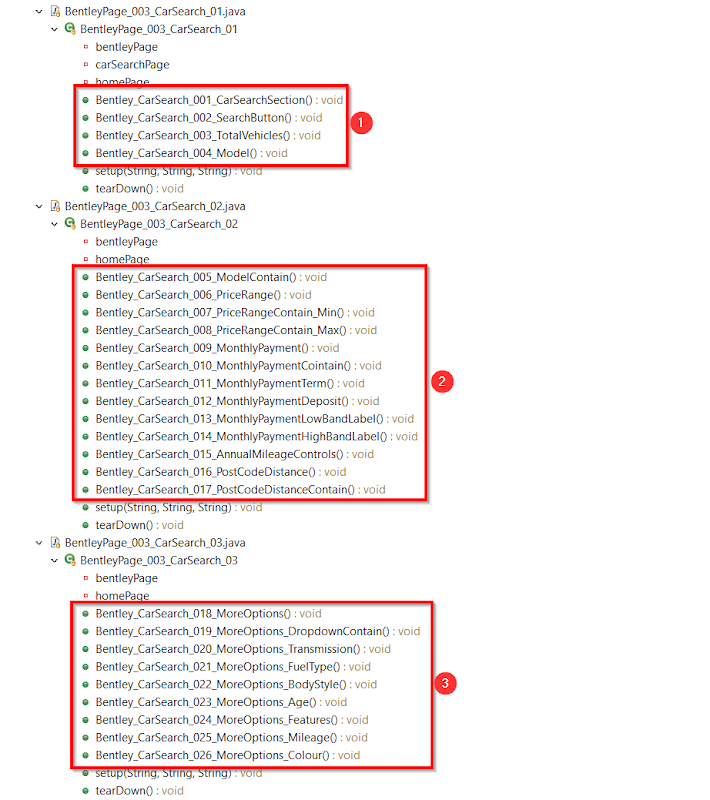
* Khi đặt tên hàm/ biến/ class/ package thì hạn chế sử dụng trùng với từ khóa của ngôn ngữ lập trình đang sử dụng



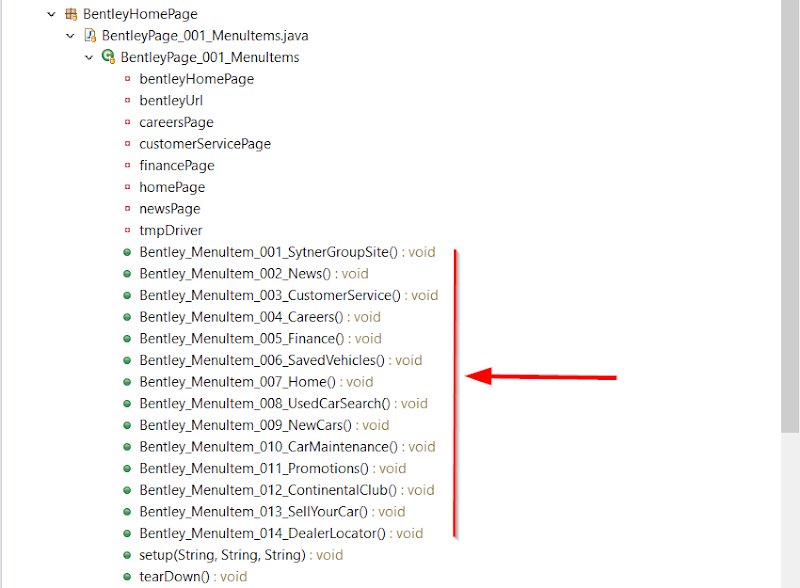


* **Package**
* Testcase (Viết thường)
* tên công ty > tên dự án > tên module
* facebook.webapp.posts
* facebook.mobile.users
* com.sytner.payment
* com.wordpress.articles
* Page Object/ PageUI (camelCase)
* pageObjects
* pageUIs
* reportConfig
* retryConfig
* ….
* **Method/ Variable name (camelCase)**
* **Hàm (Function/ Method):**
* public void clickToElement() {...}
* public void sendkeyToElement(){...}
* public String getFirstItemSelected(){...}
* **Biến (Variable/ Field):**
* String expectedValue, emailErrorMessage;
* String firstName, lastName, city, address, phoneNumber;
* int numberItem, productPrice, pageSize;
* **Class name**
* **Pascal Case**: Chữ đầu tiên của 1 từ là viết hoa (Phổ biến cho các class khác ngoài testcase)
* LoginPageObject
* AbstractPage
* **UnderScore**: Có dấu gạch nối giữa các từ (Phổ biến cho testcase)
* Login\_Page\_Object
* Abstract\_Page
* **Tầng testcases (Kĩ năng để quản lí testcase):**
* **Order theo alphabe + số thứ tự + chức năng cần test**
* Account\_01\_Register\_Login\_Logout
* Account\_02\_Validate\_Login\_Form
* Uploader\_20\_Uploader\_Match\_With\_Web\_App

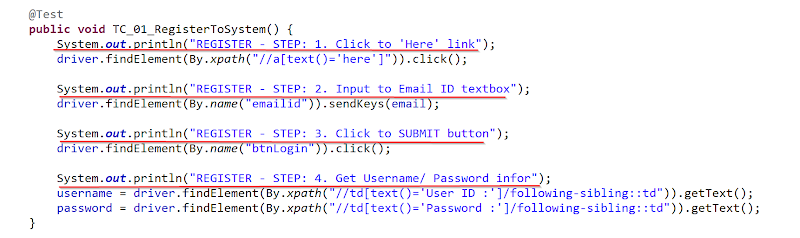




* **Tầng pageObject/ pageUIs/ commons/...:**
* Đặt tên theo chức năng của class đó (Pascal Case)
* LoginPageObject
* HomePageObject
* LoginPageUI
* HomePageUI
* AbstractPage/ BasePage
* AbstractTest/ BaseTest
* GlobalConstants
* ExtentReportListener
* ...
* **Testcase name (Pascal Case kết hợp vs Underscore)**
* **Order theo thứ tự + chức năng của testcase (Nếu như có nhiều testcase thì nên tách ra thành nhiều Class cho dễ quản lí)**
* User\_01\_RegisterToSystem
* User\_02\_LoginToSystem
* Uploader\_01\_AddFileAndCheckMatch
* Uploader\_02\_UpdateDocumentTypeAndCheckMatch
* Uploader\_03\_UploadNewFileForDocumentTypeAndCheckMatch



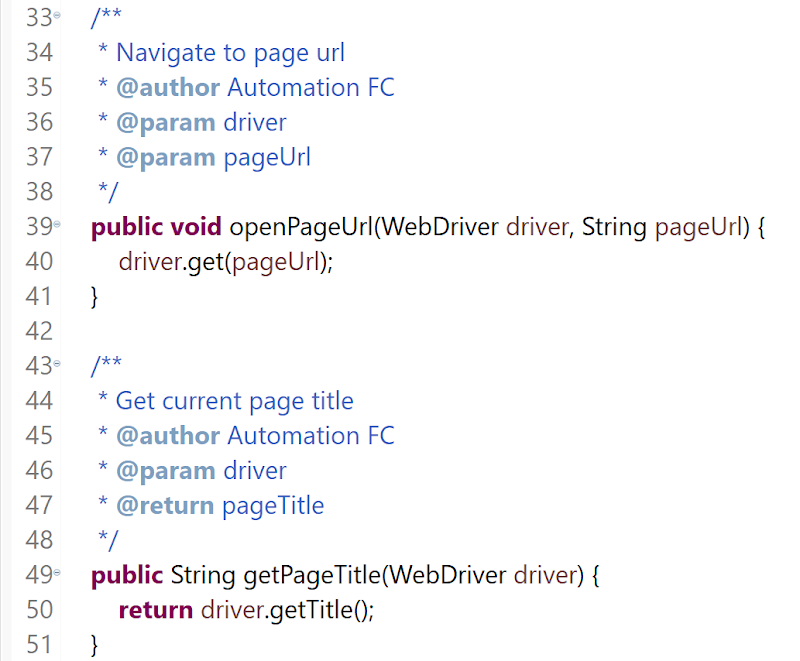
* **Không sử dụng print/ comment trong code (sau khi đã release 1 test class hoặc khi làm bài test phỏng vấn)**
* System.out.println(...);



* Comment code



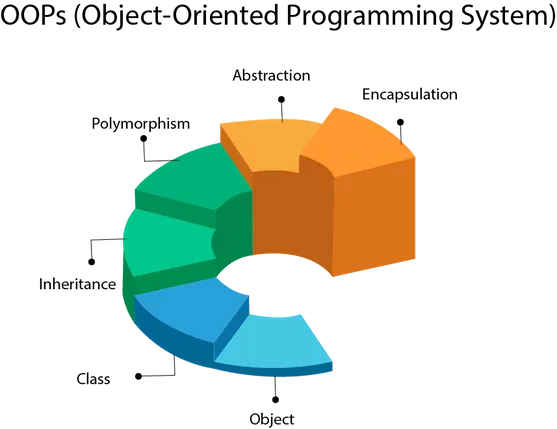
* Java Docs (Khi có 1 team/ cá nhân code cứng và build thành 1 bộ core cho các team/ cá nhân khác sử dụng)
* AbstractPage (BasePage)
* AbstractTest (BaseTest)
* PageObject (Class)
* Utilities/ Report/ Config



* **Reference:**
* [Tiêu chuẩn coding trong Java (Coding Standards)](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gpcoder.com/1775-tieu-chuan-coding-trong-java-coding-standards/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193836000%26amp;usg%3DAOvVaw1oRQ46xLVaGOECFQZn2Ufn&sa=D&source=editors&ust=1628669193893231&usg=AOvVaw27RXfF0SFRDcwCFbD6rfi6)
* [Định danh trong Java, nguyên tắc đặt tên định danh](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://stackjava.com/java-core/dinh-danh-trong-java.html%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193837000%26amp;usg%3DAOvVaw0Pf6ZRuGJ1UylEsMKZ9ruP&sa=D&source=editors&ust=1628669193893464&usg=AOvVaw1044XHKnftNa23NPeRH1vH)
* [Quy ước, chuẩn đặt tên định danh trong Java](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://stackjava.com/java-core/quy-uoc-chuan-dat-ten-dinh-danh-trong-java.html%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193837000%26amp;usg%3DAOvVaw1yVCs831up2BDe4nTSg4hj&sa=D&source=editors&ust=1628669193893615&usg=AOvVaw2Srn6gm3EgSwx-7uAx6I6U)
* [Coding-convention for Selenium-Java](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttp://pragmatictestlabs.com/2018/03/05/coding-convention-selenium-java/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193837000%26amp;usg%3DAOvVaw2uhoS5ZRWYcf_WnWKW3yNx&sa=D&source=editors&ust=1628669193893735&usg=AOvVaw2OlKPH9QlAj-k3M5ewiImt)
* [The Clean Coder - A Code of Conduct for Professional Programmers](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://drive.google.com/file/d/1bo_SqffHF1S06D_t1ZV4noMHdexM0bUG/view?usp%253Dsharing%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193837000%26amp;usg%3DAOvVaw1OjNdU45i_qUxiuFhXZSgE&sa=D&source=editors&ust=1628669193893866&usg=AOvVaw23KYq0sxjE_tZc_H5xOElk)
* [The Clean Code - A Handbook of Agile Software Craftmanship](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://drive.google.com/file/d/1LIHR7zMDcPr1VSCu5uBHuOW0-5kpo-uG/view?usp%253Dsharing%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193838000%26amp;usg%3DAOvVaw3p_6AQ4xniylLF8YjIF0zl&sa=D&source=editors&ust=1628669193893987&usg=AOvVaw1L0WP8y6a2b62jFPjz58mW)
* [Review sách: Clean Code: A Handbook of Agile Software Craftsmanship](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://toidicodedao.com/2015/04/09/review-sach-clean-code-a-handbook-of-agile-software-craftsmanship/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193838000%26amp;usg%3DAOvVaw1uRf6dBWFoRH7YpRtcTOj_&sa=D&source=editors&ust=1628669193894111&usg=AOvVaw0GCQFpZWI8AN0p6SDATFmd)

**OOP Knowledge**

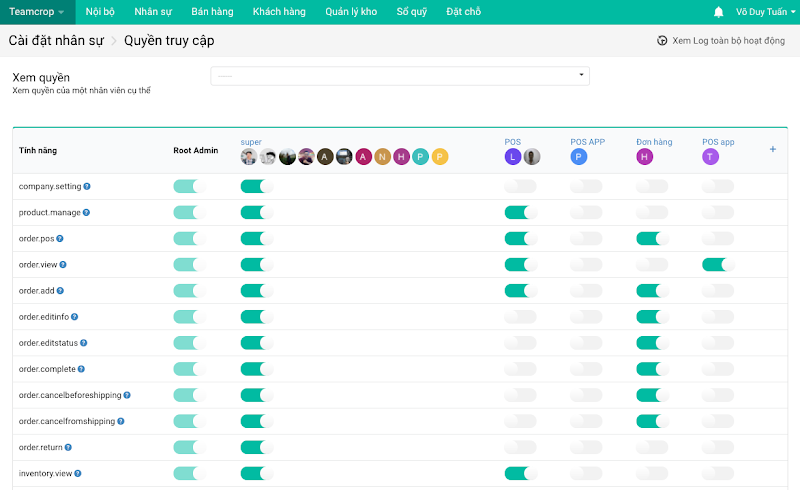
* **Các ngôn ngữ hỗ trợ OOP**
* Java
* C#
* Python
* Ruby
* C++
* Typescript
* **Lập trình hướng đối tượng (Object Oriented Programming)**
* **Đóng gói - Encapsulation:** Đóng gói các thuộc tính/ phương thức lại vào trong 1 Class để cho các Class khác kế thừa/ sử dụng - Tính đóng gói là không cho phép bên ngoài biết được bên trong đối tượng có những gì hay logic xử lí như thế nào - nếu muốn thay đổi bên trong đối tượng thì phải được sự chấp nhận của đối tượng đó thông qua các mức độ truy cập - tăng tính bảo mật và ngăn chặn việc truy cập không hợp lệ
* **Access Modifier - Phạm vi truy cập**: Cho phép 1 class khác có thể truy cập vào thuộc tính/ phương thức của class này hay không (phân quyền chức năng)
* **Kế thừa - Inheritance:**1 class con có thể sử dụng những thuộc tính/ phương thức của Class cha chia sẻ cho nó
* **Đa hình - Polymorphism:**Một hàm có thể có nhiều hành vi/ trạng thái khác nhau tùy vào ngữ cảnh gọi ra để sử dụng
* **Trừu tượng - Abstraction:**Abtract Class và Interface



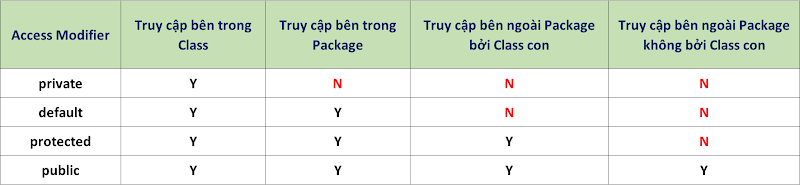
**Class and Object**

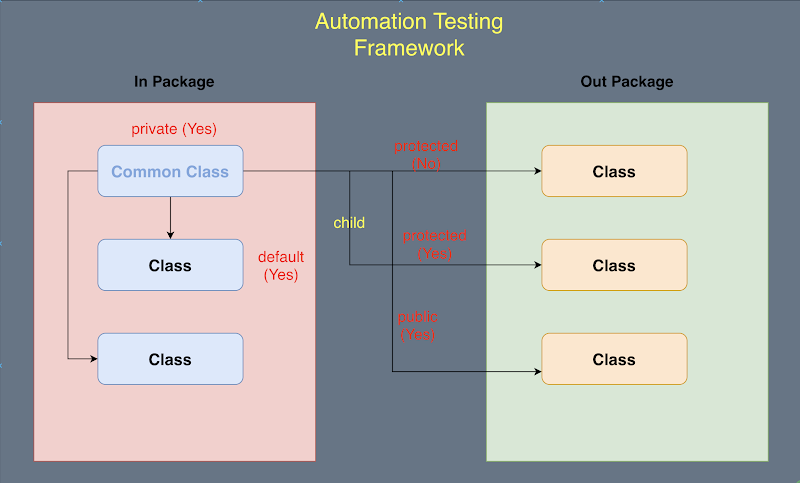
* **Class**
* public
* Được phép kế thừa hoặc khởi tạo object
* abstract
* Liên quan tới tính trừu tượng
* Được phép kế thừa
* Không được phép khởi tạo object
* final
* Không cho các class khác kế thừa lại
* Được phép khởi tạo object

**Access Modifier**



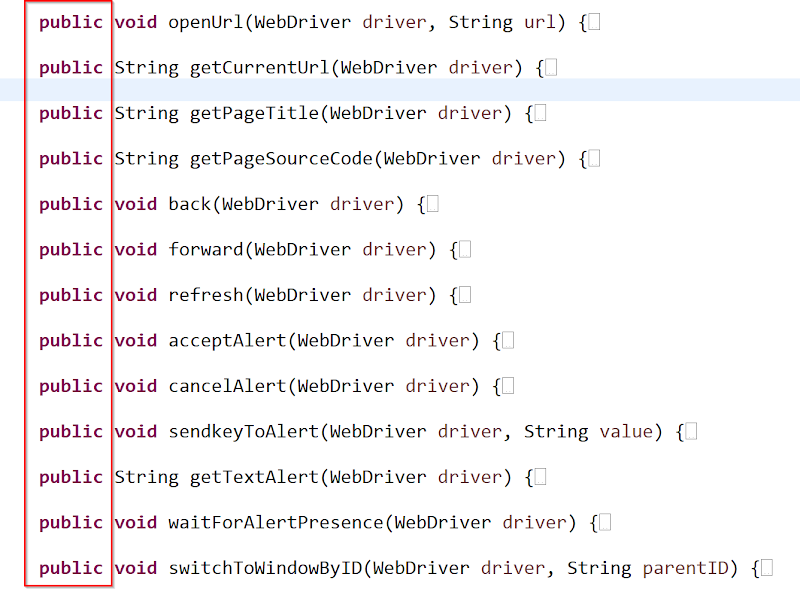
* Một trong những nguyên tắc của lập trình hướng đối tượng ([Nguyên tắc đóng gói](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://vn.got-it.ai/blog/tinh-dong-goi-trong-oop-va-nhung-dieu-ban-can-biet%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193841000%26amp;usg%3DAOvVaw3qcogGJlQ2JJI3aWXeChsx&sa=D&source=editors&ust=1628669193895242&usg=AOvVaw0O2Tr8yrDENoiVJ9PwaWQw))
* Cho phép 1 class nằm chung hoặc ngoài package truy cập vào dữ liệu của 1 class khác hay ko
* Thuộc tính (Field/ Variable)
* Phương thức (Function/ Method)
* Từ khóa
* private: màu đỏ
* default: màu xanh da trời
* protected: màu vàng
* public: màu xanh lá cây



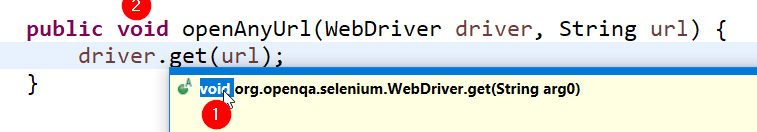


**Common Function/ Method Rules**

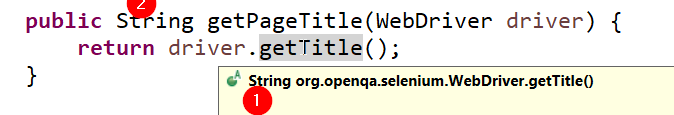
* Tại sao phải cần các common function (hàm dùng chung)
* Giúp cho code dễ bảo trì hơn
* Giúp cho code dễ mở rộng và linh hoạt hơn
* Tái sử dụng lại nhiều lần bằng cách gọi hàm
* Phạm vi set là public tại **AbstractPage/ BasePage**



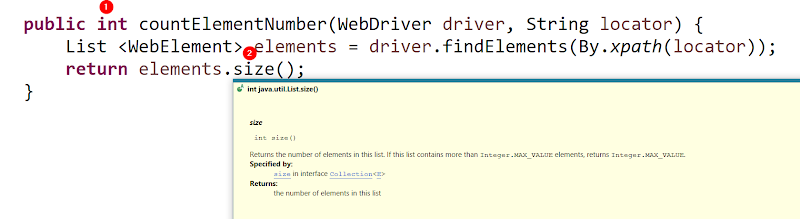
* **Method/ Function name (Tên hàm - camelCase):**
* Có nghĩa đầy đủ và dễ hiểu/ ko nên viết quá dài (để cho người đọc biết hàm này có chức năng gì mà ko cần mở code ra)
* openUrl() {...}
* getPageTitle() {...}
* clickToElement() {...}
* selectItemInDropdown() {...}
* …
* Một hàm chỉ xử lí 1 chức năng riêng biệt - không implement theo cách một hàm xử lí cho nhiều chức năng/ yêu cầu khác nhau
* [Single Responsibility Principle](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://toidicodedao.com/2016/05/03/series-solid-cho-thanh-nien-code-cung-single-responsibility-principle/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193843000%26amp;usg%3DAOvVaw1D27yeCfXwpyXi7D3KXIV0&sa=D&source=editors&ust=1628669193896374&usg=AOvVaw1SdPJkEj4aB_m4qN5e2LVH)
* **Parameter (Tham số của hàm):**
* **Web Browser:**
* Mandatory (Bắt buộc): (WebDriver driver)
* **Web Element:**
* Mandatory (Bắt buộc): (WebDriver driver, String locator)
* **Note:**
* driver: để cho nó map được driver giữa nhiều class khác nhau/ ko bị lỗi NullPointerException (Chưa khởi tạo 1 đối tượng của Class)
* locator: Để nó thao tác/ action lên cái element nào (locator là đại diện cho 1 element/ object: textbox/ textarea/ button/...)
* **Example**:
* openPageUrl(WebDriver driver, String urlValue) {...}
* clickToElement(WebDriver driver, String locator) {...}
* **Return data type (Kiểu dữ liệu trả về):**
* Kiểu dữ liệu trả về chính là kiểu dữ liệu của hàm - cùng capability
* return phải nằm ở dòng code cuối cùng trong 1 hàm
* kiểu void -> không trả về
* click/ sendkey/ selectByText/ dragAndDrop/...



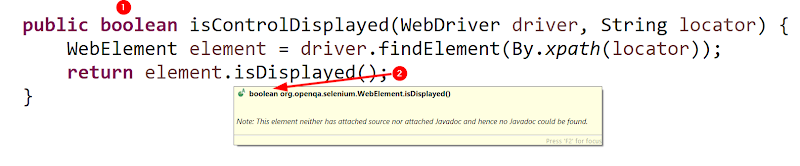
* String
* getCurrentUrl/ getTitle/ getAttribute/ getText/ getCssValue/...



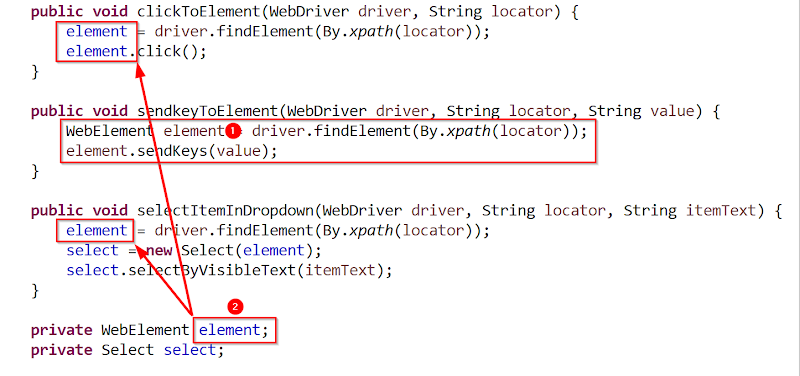
* int
* get List<WebElement> size



* boolean (true/ false)
* isDisplayed/ isEnabled/ isSelected/ isMultiple/..



* **Biến toàn cục và cục bộ**
* **Toàn cục:**Khai báo bên ngoài các hàm - bên trong class - có thể truy cập ở trong toàn bộ class đó (2)
* **Cục bộ:** Khai báo bên trong hàm - có thể truy cập ở trong hàm đó (1)



**Selenium Web Browser**

1. get(Url) -> openPageUrl -> Mở ra 1 page
2. getTitle -> String
3. getCurrentURL -> String
4. getPageSource -> String
5. back
6. forward
7. refresh
8. Alert
9. waitAlertPresence \*
10. acceptAlert
11. cancelAlert
12. getTextAlert \*
13. sendkeyToAlert \*
14. [Windows](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gist.github.com/daominhdam/dd2ea818599026d5addd557298dfac1c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193846000%26amp;usg%3DAOvVaw07pYp1l6ipXFE5nOKC_mfr&sa=D&source=editors&ust=1628669193898240&usg=AOvVaw12kKSgi7GsbiqxOhcYfy3X)
15. switchWindowByID
16. switchWindowByTitle
17. closeAllWindowsWithoutParent

**Selenium Web Element**

**Note: driver.findElement/ findElements**

1. findElement(s)

|  |
| --- |
| **public** WebElement getElement(String locator) {         **return** driver.findElement(getByXpath(locator)); }  **public** List<WebElement> getElements(String locator) {         **return** driver.findElements(getByXpath(locator)); } |

1. byXpath

|  |
| --- |
| **public** **By** getByXpath(String locator) {         **return** **By**.xpath(locator); } |

1. clickToElement
2. button
3. link
4. checkbox
5. radio button
6. dropdown
7. image
8. ...

|  |
| --- |
| **public** **void** clickToElement(**String** locator) {         element = find(locator);         element.click(); } |

1. sendkeyToElement
2. textbox
3. textarea
4. dropdown list (editable)
5. upload file
6. selectItemInDropdown
7. getSelectedItemInDropdown
8. isDropdownMultiple
9. [selectItemInCustomDropdown](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gist.github.com/daominhdam/096b5ceda7ced2a1d0047bfa8969c3df%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193850000%26amp;usg%3DAOvVaw24Tv5ZsT6YbRm2yliz1dX8&sa=D&source=editors&ust=1628669193899636&usg=AOvVaw0aoIaAZa6FNtQRL9-l-PIX)
10. getAttributeValue
11. getTextElement
12. getCssValue
13. convertRgbaToHexa

|  |
| --- |
| public String covertRgbaToHex(String rgbaValue) {         return Color.fromString(rgbaValue).asHex(); } |

1. getElementSize(**findElements**)
2. checkTheCheckboxOrRadio
3. uncheckTheCheckbox
4. isControlDisplayed
5. isControlSelected
6. isControlEnabled
7. Frame/ Iframe
8. switchToIframe/ Frame
9. switchToDefaultContent
10. User Actions
11. doubleClickToElement
12. hoverMouseToElement **-> moveToElement**
13. rightClick
14. dragAndDrop
15. sendKeyboardToElement

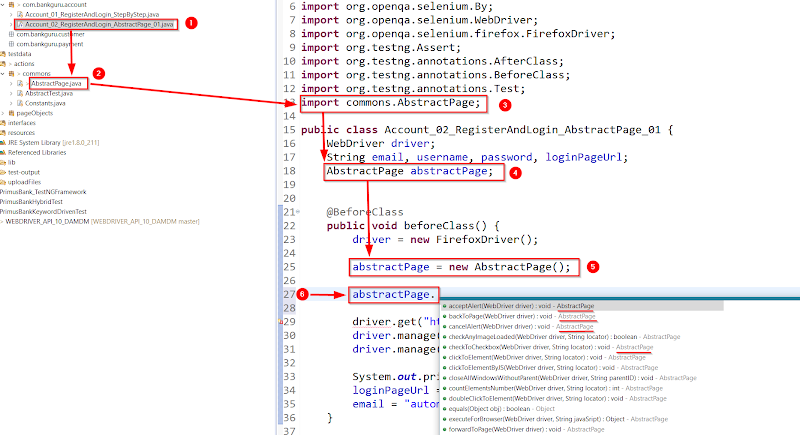
**Note: required “perform()” action for all above functions**

|  |
| --- |
| public void sendKeyboardToElement(WebDriver driver, String locator, Keys key) {         element = driver.findElement(By.xpath(locator));         action = new Actions(driver);         action.sendKeys(element, key).perform(); } |

1. Upload (Update trong bài Upload File - Build Framework)
2. sendkey
3. 1 file
4. multi-file \*
5. robot
6. autoIT
7. [JavascriptExecutor](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gist.github.com/daominhdam/10656ee5d960c9d83bbabc0caee0cb7b%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193853000%26amp;usg%3DAOvVaw35Vb1hAs-MxqRTiqiudcle&sa=D&source=editors&ust=1628669193900949&usg=AOvVaw3kEE6qZRWENIIr3rKfJ-kT) \*
8. executeJavascriptToBrowser
9. executeJavascriptToElement
10. scrollToBottomPage
11. scrollToElement
12. hightlightElement
13. removeAttributeOfElement
14. checkImageLoaded
15. verifyTextInInnerText
16. [getElementValidationMessage](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://automationfc.com/2018/10/15/java-webdriver-14-verify-html5-validation-message/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193854000%26amp;usg%3DAOvVaw0_uiQyS1gRoeYRf_pT0ryT&sa=D&source=editors&ust=1628669193901350&usg=AOvVaw0mS3nwr2tYsJdr0gVRM-u3)
17. waitToJQueryAndJSLoadedSuccess
18. Wait
19. **waitForElementVisible** \*
20. **waitForElementClickable**\*
21. **waitForElementInvisible** \*
22. **waitForAlertPresence** \*

**Use AbstractPage (BasePage)**

* **01 - Import + Declare + Init AbstractPage class**
* Phải khai báo
* Phải khởi tạo
* Thông qua instance (đại diện) của class AbstractPage (BasePage)
* Giống như cách sử dụng các thư viện/ class thông thường của phần học Selenium API
* WebDriver driver = new FirefoxDriver();
* Actions action = new Actions(driver);
* Select select = new Select(..);
* WebDriverWait explicitWait = new WebDriverWait(driver, timeout);
* ...



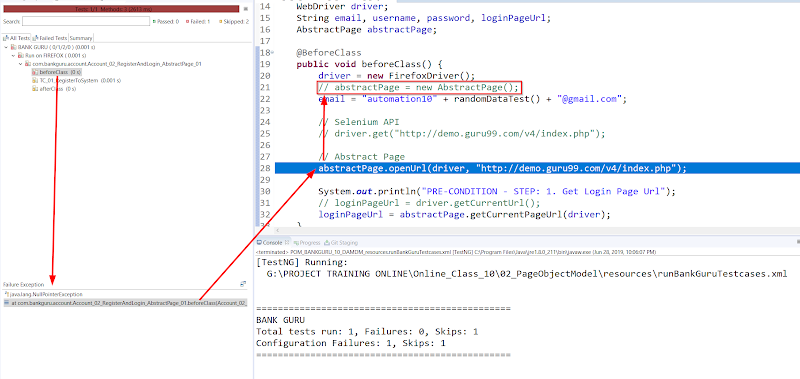
* **02 - Static method**
* Áp dụng nguyên tắc**Đóng gói - Encapsulation** (Che giấu/ ẩn giấu sự khởi tạo của 1 đối tượng/ Class)

|  |
| --- |
| *// BasePage* public static BasePage getBasePage() {                 return new BasePage(); }  *// Test Class* BasePage basePage;  basePage = BasePage.getBasePage(); |

* **03 - Extends AbstractPage**
* 1 trong những nguyên tắc của lập trình OOP
* **Kế thừa - Inheritance**
* Không cần khai báo
* Không cần khởi tạo
* Không cần thông qua instance - gọi **trực tiếp các thuộc tính/ phương thức** trong AbstractPage class



* **NOTE:** Nếu không khởi tạo theo cách 1 thì bị lỗi NullPointerException (Chưa khởi tạo class mà sử dụng các thuộc tính/ phương thức của class khác)



**References**

* [Lập trình hướng đối tượng (OOPs) trong java](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gpcoder.com/2227-lap-trinh-huong-doi-tuongoops-trong-java/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193858000%26amp;usg%3DAOvVaw1kbE73f9tSoLqfnp3-0DPc&sa=D&source=editors&ust=1628669193902909&usg=AOvVaw1ECgra36hNwgwlN1AC_pNS)
* [Từ khóa this và super trong Java](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gpcoder.com/2260-tu-khoa-va-super-trong-java/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193858000%26amp;usg%3DAOvVaw3wtQKASgp04FS6PupeNfu2&sa=D&source=editors&ust=1628669193903050&usg=AOvVaw1034kZRkQxn0uuTTGR2zDx)
* [Tính đóng gói (Encapsulation) trong java](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gpcoder.com/2365-tinh-dong-goi-encapsulation-trong-java/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193859000%26amp;usg%3DAOvVaw22xDw0jAkvBV4VJmv9PxJw&sa=D&source=editors&ust=1628669193903149&usg=AOvVaw0bOdTeC9Jr5wXyG59pwjQB)
* [Tính kế thừa (Inheritance) trong java](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gpcoder.com/2376-tinh-ke-thua-inheritance-trong-java/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193859000%26amp;usg%3DAOvVaw3tbnzY1BKO9uzi25hzjLb1&sa=D&source=editors&ust=1628669193903254&usg=AOvVaw2bFJTNrvpyyLmwPA7z_p5S)
* [Tính đa hình (Polymorphism) trong Java](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gpcoder.com/2386-tinh-da-hinh-polymorphism-trong-java/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193859000%26amp;usg%3DAOvVaw03aUG1KfjQkBtQYdSoVecY&sa=D&source=editors&ust=1628669193903355&usg=AOvVaw1QlnDTylrPLQbNLbLw2cLx)
* [Tính trừu tượng (Abstraction) trong Java](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gpcoder.com/2388-tinh-truu-tuong-abstraction-trong-java/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193860000%26amp;usg%3DAOvVaw1JTuLO6SDmKV1H2JvmYTXp&sa=D&source=editors&ust=1628669193903459&usg=AOvVaw388Ttxq5LxZ566R-r-NAJ6)
* [Các nguyên lý thiết kế hướng đối tượng - SOLID](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://gpcoder.com/4200-cac-nguyen-ly-thiet-ke-huong-doi-tuong/%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1628669193860000%26amp;usg%3DAOvVaw0p8VyyO1-mFyz_trkKa_0n&sa=D&source=editors&ust=1628669193903584&usg=AOvVaw2G9EVzlaD-O5a2dZ7MMSY0)