**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**KHOA CÔNG NGHỆ TT & TRUYỀN THÔNG**



**BÁO CÁO KẾT QUẢ**

**MÔN BẢO TRÌ PHẦN MỀM**

**ĐỀ TÀI**

**Tìm hiểu công cụ đảo ngược dùng trong bảo trì phần mềm**

**NHÓM 1**

Lương Đức Duy

Ngô Minh Phương

Hồ Hữu Nhân

Trần Thanh Điền

**Cần Thơ 8/2014**

Mục lục

[Kế hoạch thực hiện iii](#_Toc399615414)

[1 Giới thiệu 1](#_Toc399615415)

[2 Lịch sử ra đời 1](#_Toc399615416)

[3 Các đặc điểm chính 2](#_Toc399615417)

[3.1 Chuyển code sang sơ đồ UML 2](#_Toc399615418)

[3.1.1 Class diagram 2](#_Toc399615419)

[3.1.2 Sequence diagram 14](#_Toc399615420)

[3.2 Chuyển code sang Flowchart 18](#_Toc399615421)

[3.2.1 Chuyển code sang flowchart, mã giả 18](#_Toc399615422)

[3.2.2 Đồng bộ giữa mã giả, flowchart với code. 18](#_Toc399615423)

[3.2.3 Kéo thả giữa các thành phần có sẵn 19](#_Toc399615424)

[3.2.4 Chức năng “High light” cho phép người dùng quan sát code lẫn sơ đồ rõ ràng hơn. 19](#_Toc399615425)

[3.2.5 Xuất tài liệu 19](#_Toc399615426)

[3.2.6 Ưu và nhược điểm 21](#_Toc399615427)

# Kế hoạch thực hiện

|  |  |
| --- | --- |
| **Thời gian** | **Nội dung** |
| Tuần 1 – Tuần 4 | Các thành viên về tìm hiểu 1 công cụ đảo ngược.  Họp nhóm các thành viên báo cáo và phản biện. |
| Tuần 5 | Thống nhất chọn công cụ theo hướng IDE tích hợp plugin.  Tiếp tục nghiên cứu và viết báo cáo dựa trên công cụ.  Lương Đức Duy, Ngô Minh Phương:   * Nghiên cứu plugin eUML 2 dùng để chuyển code về Sequence Diagram và Class diagram. * Mức độ hoàn thành: 100%.   Trần Thanh Điền, Hồ Hữu Nhân:   * Nghiên cứu plugin CodeRocket dùng để chuyển code về Flowchart. * Mức độ hoàn thành: 100%. |
| Tuần 6 | Review tài liệu.  Chuẩn bị báo cáo + demo thử trước nhóm.  Công việc:   * Lương Đức Duy: Chuẩn bị slide báo cáo và review tài liệu lần 1. * Lương Đức Duy, Ngô Minh Phương: Chuẩn bị demo phần chuyển mã nguồn sang sơ đồ UML. * Hồ Hữu Nhân, Trần Thanh Điền: Chuẩn bị demo phần chuyển code sang flowchart, mã giả. |
| Tuần 7 | Review tài liệu.  Các thành viên tiếp tục demo thử và hoàn thiện báo cáo. |

# Giới thiệu

Công cụ đảo ngược có vai trò chiết xuất thông tin về kiến thức hoặc thiết kế của phần mềm từ mã nguồn có sẵn. Công cụ đảo ngược giúp nhóm bảo trì hiểu rõ được phần mềm mình cần bảo trì theo nhiều mục tiêu khác nhau. Công cụ đảo ngược có thể trích xuất mô hình thiết kế UML thiết kế cho phần mềm đang cần bảo trì, có thể chuyển các đoạn mã thành sơ đồ giải thuật, hiểu rõ hơn cơ sở dữ liệu, v.v Mục tiêu cuối cùng của công cụ đảo ngược hiểu rõ, hiểu nhanh được phần mềm mình cần bảo trì. Đặc biệt trong các chương trình, hệ thống phức tạp, hoặc trong trường hợp tài liệu của sản phẩm đã lỗi thời hoặc bị mất.

Nhóm chọn nghiên cứu theo hướng IDE, cụ thể là Eclipse (mục tiêu trên ngôn ngữ Java) tích hợp các plugin để thực hiện quá trình đảo ngược.

Lý do:

* Eclipse miễn phí.
* Eclipse là IDE lập trình mã nguồn mở rất phổ biến hiện nay.
* Eclispe bản thân là một IDE nên lập trình viên có thể vừa đảo ngược phần mềm, vừa có thể tiếp tục bảo trì phần mềm trực tiếp trên Eclipse.
* Do tính mở nên có rất nhiều công ty đã tùy chỉnh Eclipse theo những công cụ nhất định, tuy nhiên do phát triển nên từ Eclipse nên các công cụ khác đều thừa hưởng được kho plugin phong phú mà Eclipse hỗ trợ.
* Do Eclipse bản thân cũng là công cụ mã nguồn mở nên có khả năng tùy chỉnh rất cao, chúng ta có thể tự viết plugin cho Eclipse.
* Eclipse hỗ trợ tốt Java và các Framework hỗ trợ Java.
* Eclipse được phát triển bởi tổ chức Eclipse và được sự hỗ trợ từ cộng đồng mạng rất lớn.

Bài báo cáo sẽ nói về lịch sử của Eclipse và giới thiệu plugin tích hợp vào Eclipse giúp Eclipse trở thành công cụ đảo ngược mạnh mẽ.

# Lịch sử ra đời

Eclipse ban đầu được phát triển như là dự án Smart Canada, phục vụ cho dòng máy tính IBM VisualAge. Trong tháng mười một năm 2001, một tập đoàn được thành lập, có hội đồng quản trị quản lý để tiếp tục phát triển Eclipse như là một phần mềm mã nguồn mở. Người ta ước tính rằng IBM đã đầu tư gần đến $40 triệu để phát triển Eclipse vào thời điểm đó. Các thành viên ban đầu trong hội đồng gồm có: Borland, IBM, Merant, QNX Software Systems, Rational Software, Red Hat, SuSE, TogetherSoft và WebGain. Vào cuối năm 2003, số thành viên trong hội đồng đã tăng lên hơn 80. Tháng một 2004, tổ chức Eclipse được thành lập.

Lịch sử phát hành:

Từ năm 2006, tổ chức Eclipse đã phối hợp phát hành các phiên bản vào tháng 6 hằng năm. Cụ thể:

|  |  |  |  |
| --- | --- | --- | --- |
| Version Name | Date | Platform Version | Projects |
| Callisto | 30/06/2006 | 3.2 | Callisto |
| Europa | 29/06/2007 | 3.3 | Europa |
| Ganymede | 25/06/2008 | 3.4 | Ganymede |
| Galileo | 24/06/2009 | 3.5 | Galileo |
| Helios | 23/06/2010 | 3.6 | Helios |
| Indigo | 22/06/2011 | 3.7 | Indigo |
| Juno | 27/06/2012 | 3.8 và 4.2 | Juno |
| Kepler | 26/06/2013 | 4.3 | Kepler |
| Luna | 25/06/2014 | 4.4 | Luna |
| Mars | 24/06/2015 (dự kiến) | 4.5 | Mars |

# Các đặc điểm chính

## Chuyển code sang sơ đồ UML

Phần này sử dụng plugin eUML2 để hỗ trợ vì ngoài việc đảo ngược bình thường, eUML hỗ trợ cho các Framework Java như: Spring, Hibernate.

### Class diagram

Về mặt cơ bản, hỗ trợ sinh sơ đồ theo 3 mức:

1. Toàn bộ dự án.
2. Package.
3. Class.

Ngoài ra, còn một số đặc điểm rất hữu ích hỗ trợ:

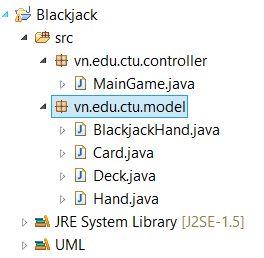
1. Real-Time Synchronization.
2. Refactoring (Tái cấu trúc)
3. Outline
4. Preferences

**Toàn bộ dự án (Project)**

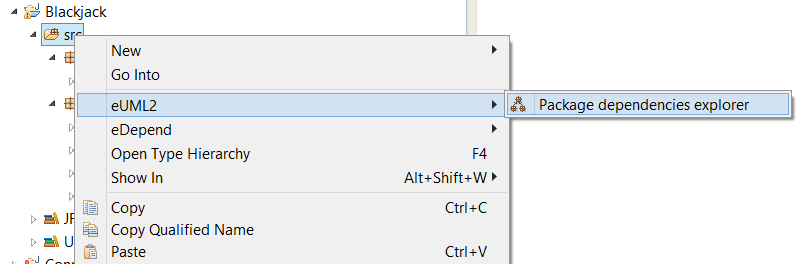
1. Chọn project.
2. Edit 🡪 eUML2.
3. Chọn Package dependence explorer.
4. Nhấn đúp vào sơ đồ Package mà muốn chuyển sang sơ đồ **Class diagram**.
5. Trong hộp thoại **Diagram options** có các tùy chọn:

* Wire automation: Association, Inheritance, Dependency.
* Scope options: Project, Package, All.

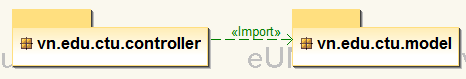
1. Nhấn OK. Sau đó xuất hiện hộp thoại **Package Content selection**.
2. Chọn class mà muốn chuyển sang **Class diagram.**
3. Chọn OK.

**Ví dụ với project BlackJack. Đây là project Java, chơi trò đánh bài Blackjack(xì tố).**

1. Chọn project. Vào Edit 🡪 eUML2 🡪Package dependence explorer.



1. Chương trình tạo ra sơ đồ lớp tương ứng trên mức toàn bộ project, cụ thể là hiển thị các package và mối quan hệ của chúng trong project.

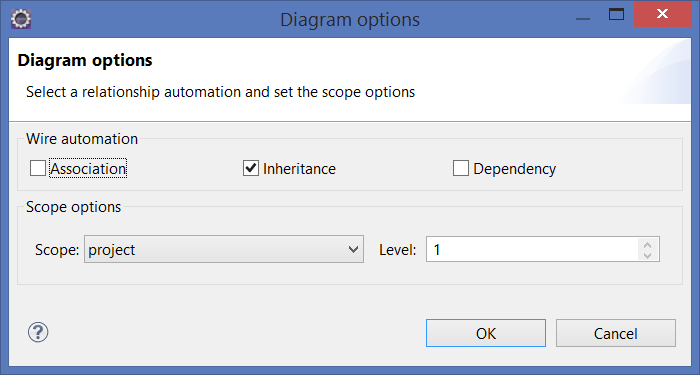


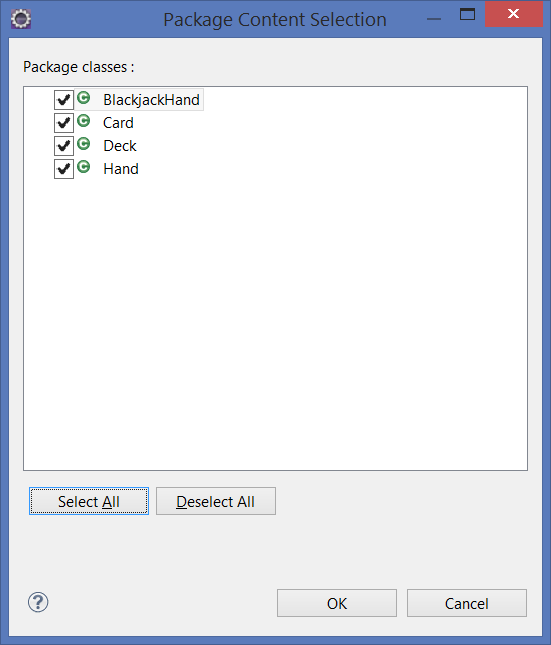
Nhấp đúp

Mối quan hệ giữa 2 package

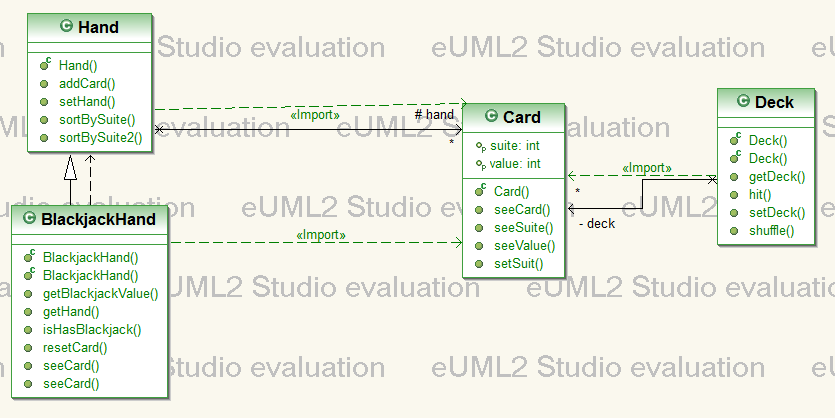
Nhấn đúp vào package bất kỳ trong Package Diagram để xem nội dung sơ đồ lớp với package tương ứng.

1. Trong hộp thoại **Diagram Options** phần Wire Automation bạn có thể chọn sinh sơ đồ theo các kiểu quan hệ giữa các Class (Association,Inheritance, Dependency) mà bạn muốn xem (ở ví dụ này chọn cả 3). Trong **Scope Options** có thể chọn 1 trong 3 phạm vị :Project, Package, All.



1. Sau khi nhấn OK, sẽ xuất hiện hộp thoại **Package Content selection**  bạn chọn những class nào mà muốn hiển thị ra ngoài class diagram.

**Kết quả: sơ đồ lớp tương ứng với các lớp vừa chọn.**



**Package.**

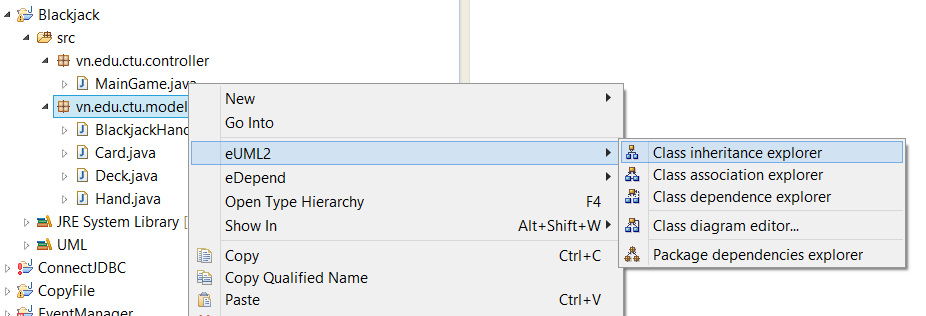
Sinh ra sơ đồ lớp trên mức package

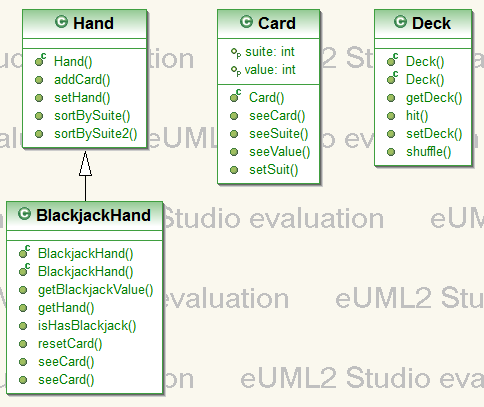
1. Chọn **Package**.
2. Edit 🡪 eUML2.
3. Có các tùy chọn:
   * Class inheritance explorer.
   * Class association explorer.
   * Class dependence explorer.
   * Class diagram editor…
   * Package depencence explorer.

**Ví dụ:** Chức năng **Class inheritance explorer**.

Cho phép sinh ra sơ đồ lớp dựa trên các mối quan hệ kế thừa của các lớp trong package.

Chọn package. Vào Edit 🡪 eUML2 **🡪** Class inheritance explorer.

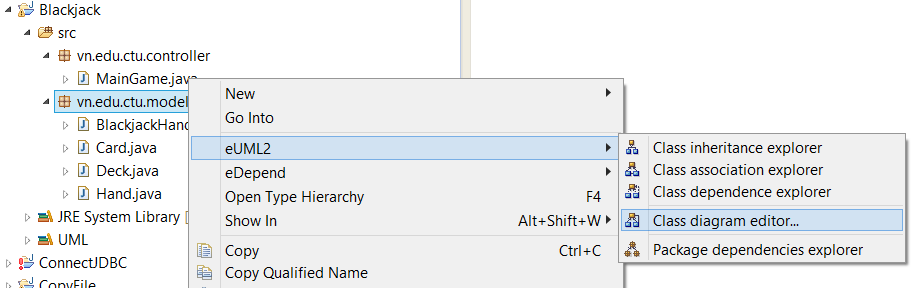


**Kết quả sinh ra mối quan hệ kế thừa giữa các lớp. Lớp Card và Deck không có mối quan hệ kế thừa nên không có mối quan hệ được sinh ra.**

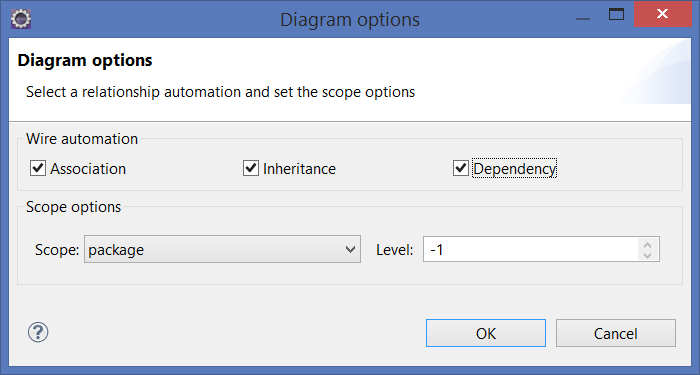
**Ví dụ**: Chức năng **Class diagram editor…**

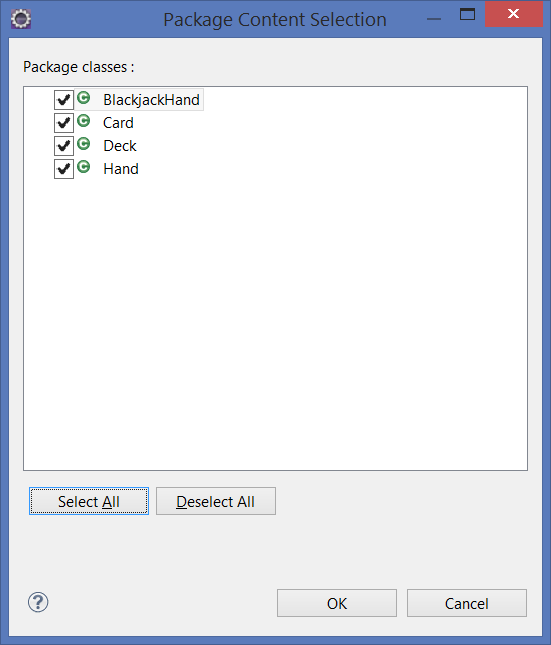
Tùy chọn mở rộng, kết hợp giữa 3 chức năng: inheritance, association, dependence.

1. Chọn package. Vào Edit 🡪 eUML2 **🡪** Class inheritance explorer.

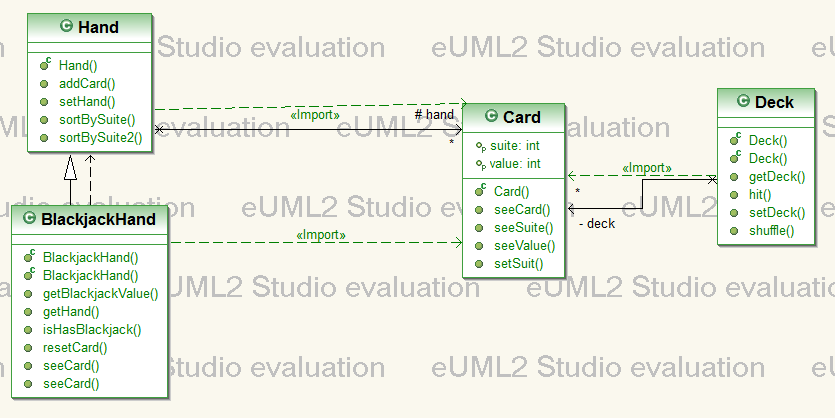


1. Sau đó sẽ xuất hiện hộp thoại “Diagram options”. Ở trong phần Wire automation chọn những tùy chọn mà bạn muốn. Sau đó bấm OK.



1. Tiếp theo sẽ xuất hiện “Package Content Selection”. Hãy chọn những class mà chúng ta muốn chuyển. Sau đó tiếp tục bấm OK.

**Kết quả là sơ đồ lớp được sinh ra dựa trên 3 kiểu quan hệ trong mục Wire Automation.**

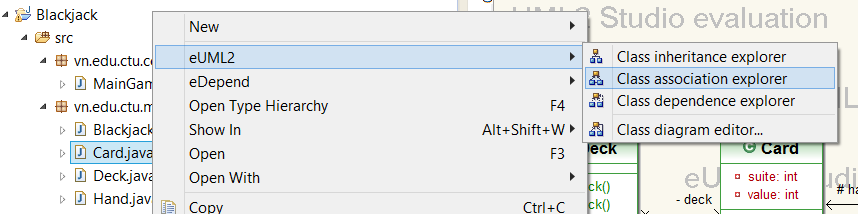


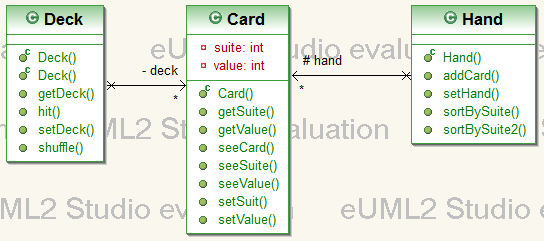
**Class**

Việc sinh ra sơ đồ cũng có các tùy chọn tương tự đối với sinh sơ đồ dựa trên package, tức là sinh ra sơ đồ UML với các mối quan hệ: inheritance, association, dependence. Tuy nhiên, chương trình chỉ phân tích dựa trên lớp cụ thể mà mình đang quan tâm thay vì toàn bộ package.

**Ví dụ:**

1. Chọn một class và nhấn chọn Edit 🡪 eUML2 🡪Class association explorer.



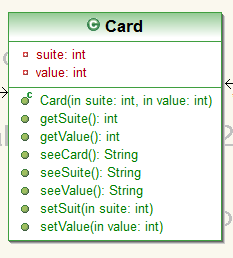
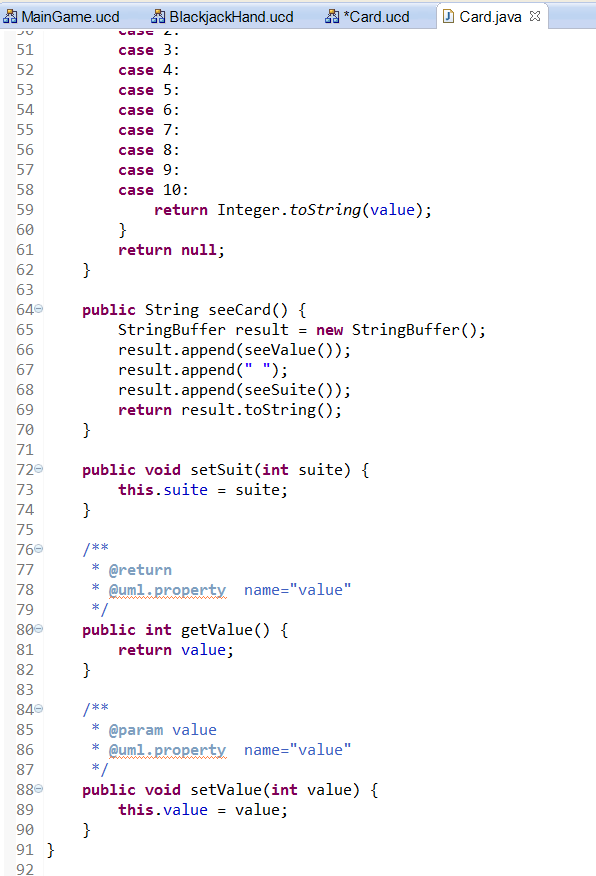
**Kết quả sinh ra là sơ đồ lớp của lớp AgreementStatusDAO và các lớp có mối quan hệ với lớp này, cụ thể là lớp Card và các mối quan hệ.**

**Các đặc điểm khác (Preferences)**

**Real-Time Synchronization**

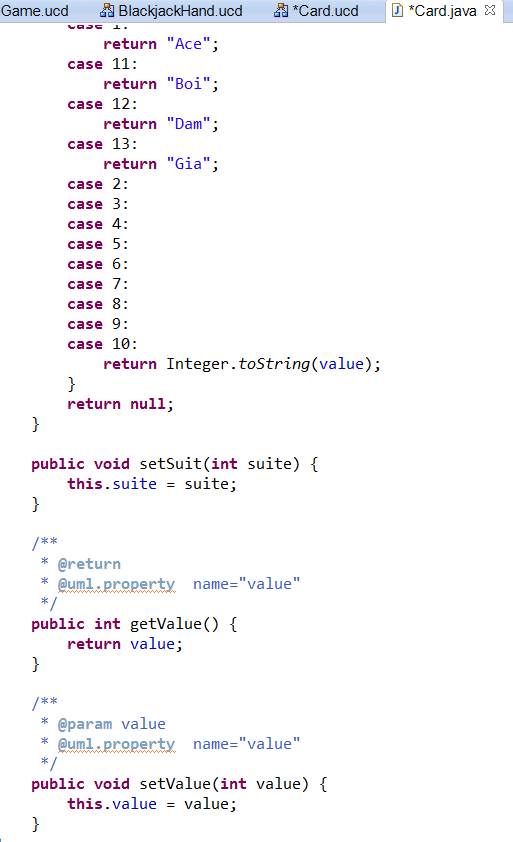
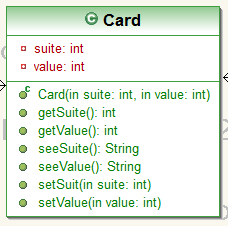
eUML2 là một plugin được tích hợp đầy đủ trong Eclipse Framework. Bạn có thể sữa code trực tiếp trên phần code java hoặc ngược lại trên sơ đồ UML và eUML2 sẽ đồng bộ ngay làm tức (Live synchronization).

**Ví dụ: lớp Card**



Xóa

Phương thức seeCard đã bị xóa.



**Refactoring (Tái cấu trúc)**

Cho phép sắp xếp thuận tiện cho việc cấu trúc lại class diagram.

Để truy cập vào các chức năng này:

1. Chọn một phần tử trong Class diagram.
2. Nhấp chuột phải để mở menu popup
3. Triển khai các menu con cấu trúc lại.

Các chức năng chính như:

* Pull up (Đưa một phương thức hoặc thuộc tính lên class cha).
* Push Down (Đưa một phương thức hoặc thuộc tính xuống class con).

|  |
| --- |
| **Trước** |
|  |

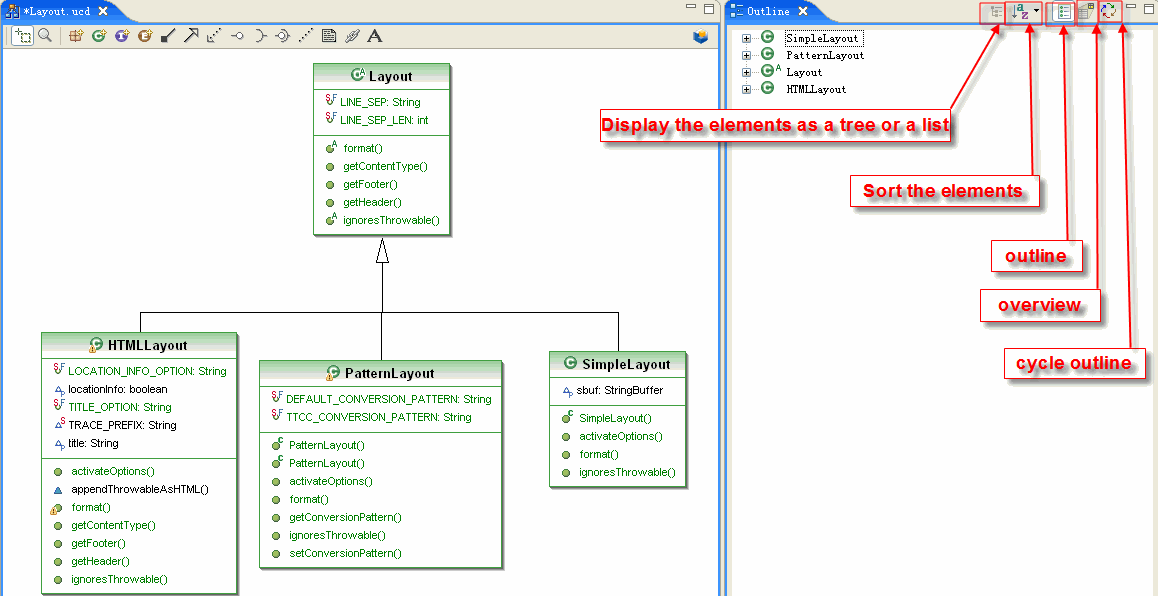
|  |
| --- |
| **Sau** |
|  |

**Outline**

Trong class diagram thì outline gồm có 3 dạng cơ bản:

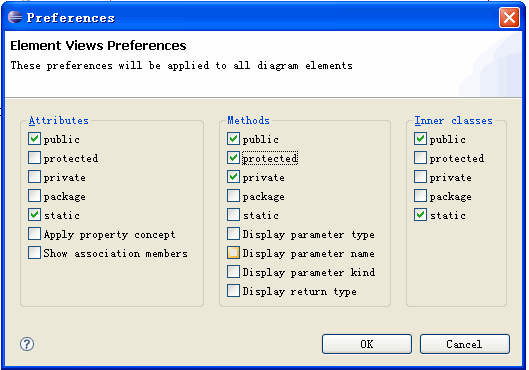
* Outline (Coi được cấu trúc của phần code, chi tiết nhứng thuộc tính hay phương thức)
* Overview (Chia thành nhiều phần View nhỏ khi mà project quá lớn)
* Cycle outline

Giúp cho người sử dụng có thể xem và quản lý phần code một cách nhanh chóng hơn.



**Element Views**

Cho phép bật tắc các thành phần hiển thị trên sơ đồ để có nhiều góc nhìn về sơ đồ đang muốn xem.

1. Chọn một class bất kỳ trong class diagram.
2. Edit 🡪 chọn Preferences
3. Chọn **Element views (**Chức năng này giúp chúng ta quản lý được các thuộc tính cũng như phương thức sẽ hiển thị trên class diagram**).**

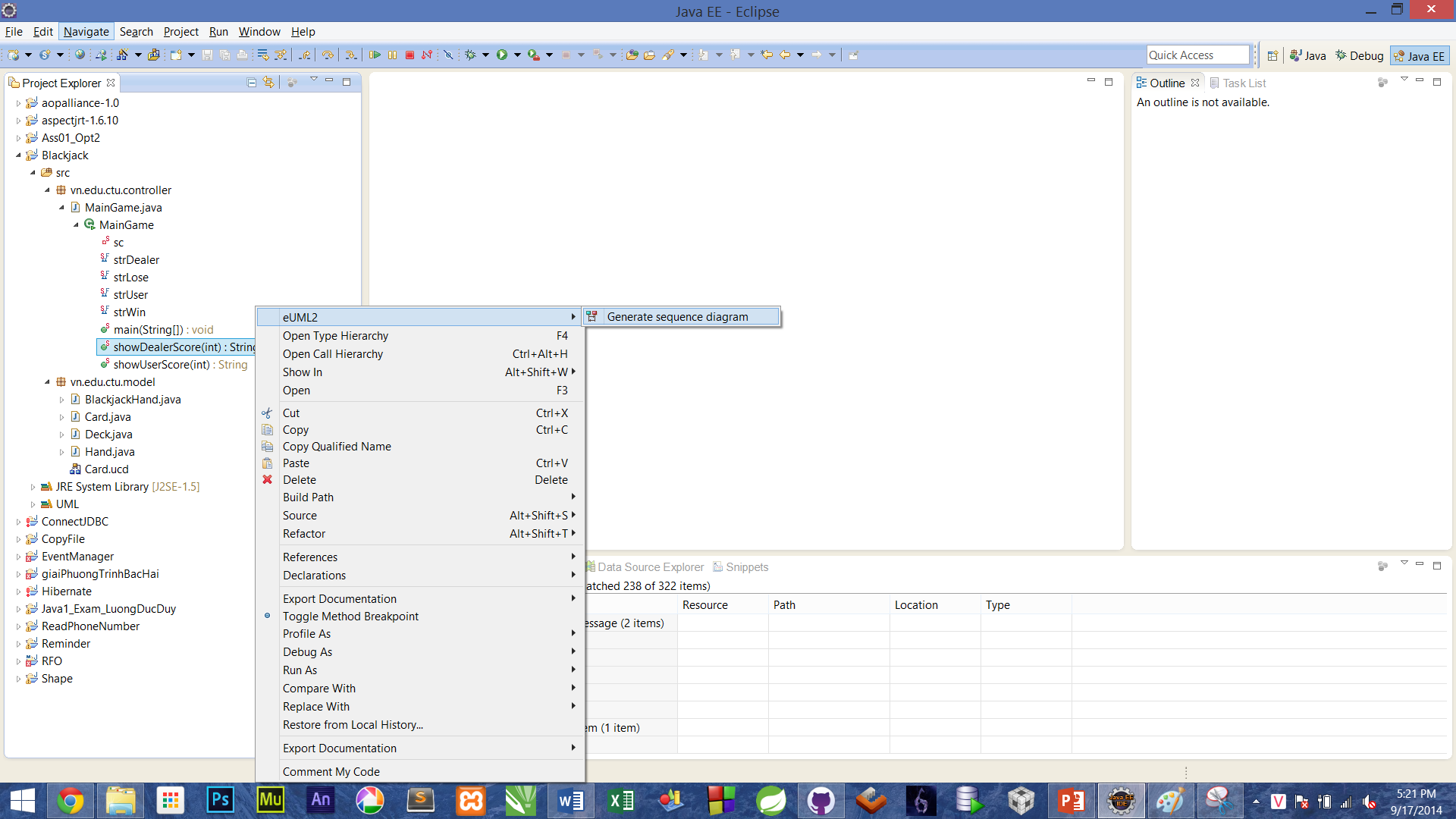
### Sequence diagram

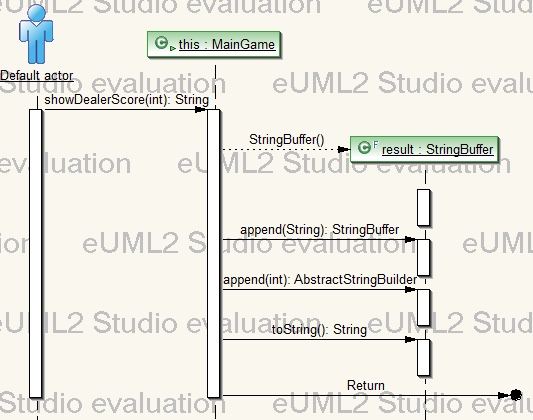
Các bước chuyển code sang sequence diagram:

1. Chọn class muốn chuyển.
2. Bấm vào biểu tượng mở rộng 
3. Chọn method (phương thức, hàm) muốn chuyển.
4. Vào Edit 🡪 eUMl2.
5. Chọn Generate sequence diagram.
6. Trong **Message filtering dialog** thì chọn Select all. Rồi bấm OK

**Ví dụ:**

1. Đầu tiên vào chọn class\method (**showDealerScore**) mà mình muốn chuyển. Edit 🡪 chọn eUML2 🡪 Generate sequence diagram.

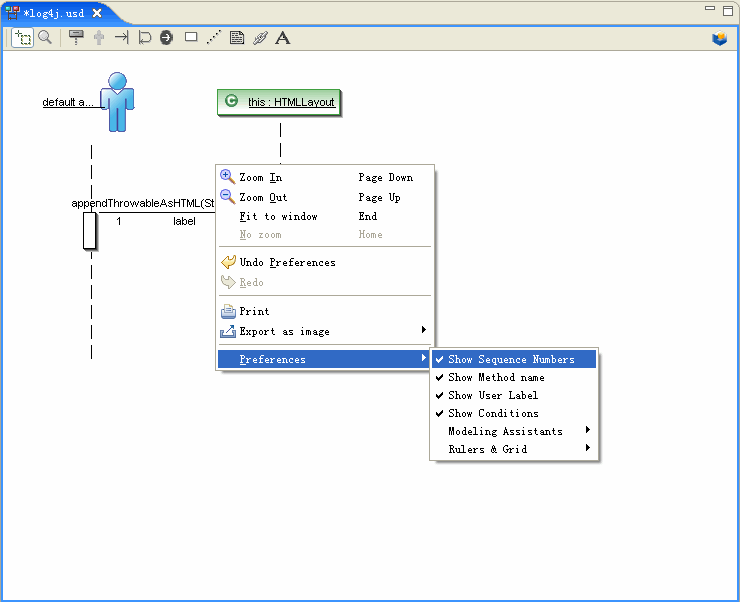


Kết quả là sơ đồ trình tự đối với phương thức **showDealerScore**.

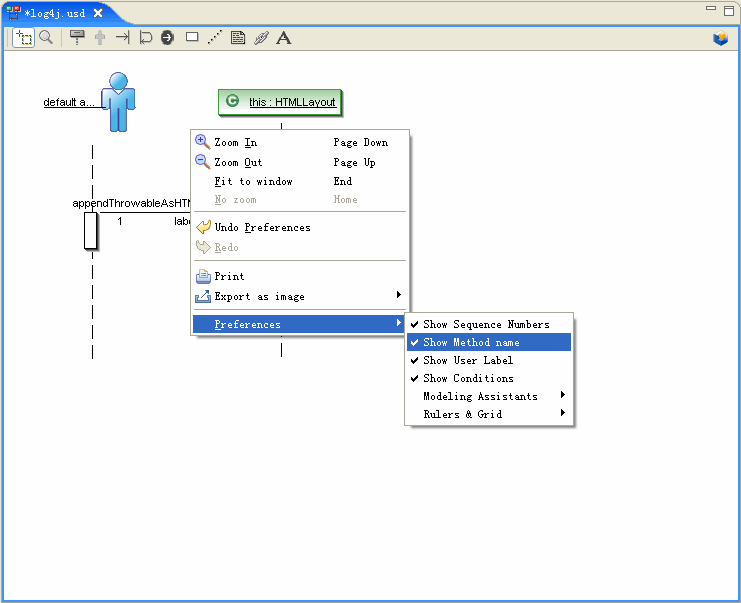
**Các đặc điểm khác (Preferences)**

Tương tự như class diagram thì sequence diagram hỗ trợ những tùy chọn như sau:

**Show Sequence Numbers**

Bật/ tắt hiển thị số thứ tự các dòng sự kiện trong sequence diagram.

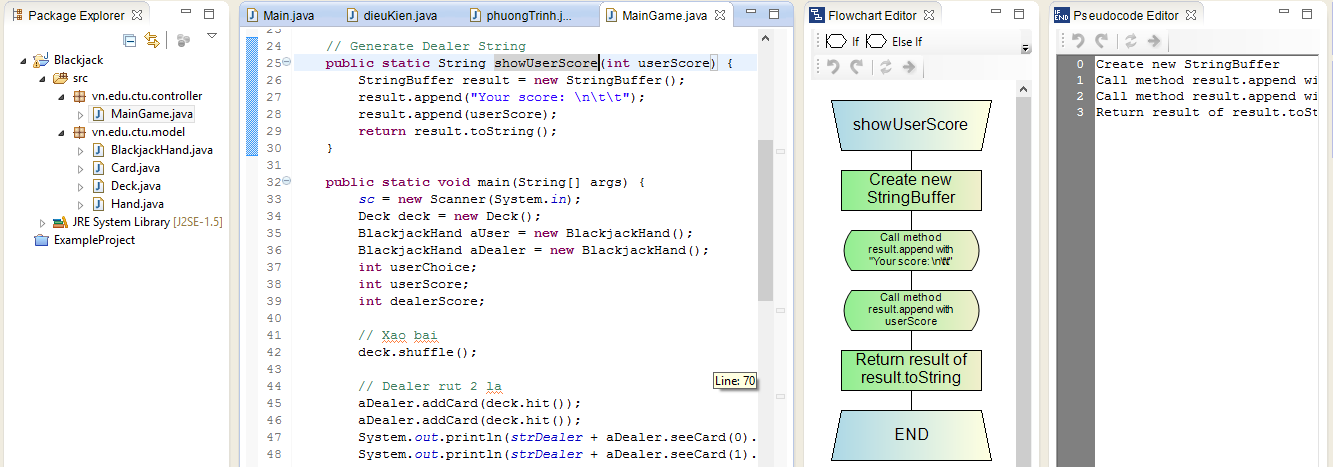
**Show Method names**

Bật/tắt hiển thị tên phương thức của dòng sự kiện.

## Chuyển code sang Flowchart

Phần này sử dụng CodeRocket đễ hỗ trợ.

### Chuyển code sang flowchart, mã giả



4. Mã giả tự động được chuyển

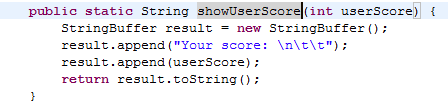
3. Flow chart tự động được chuyển

2. Click chọn hàm shoUserScore

1. Mở file cần chuyển

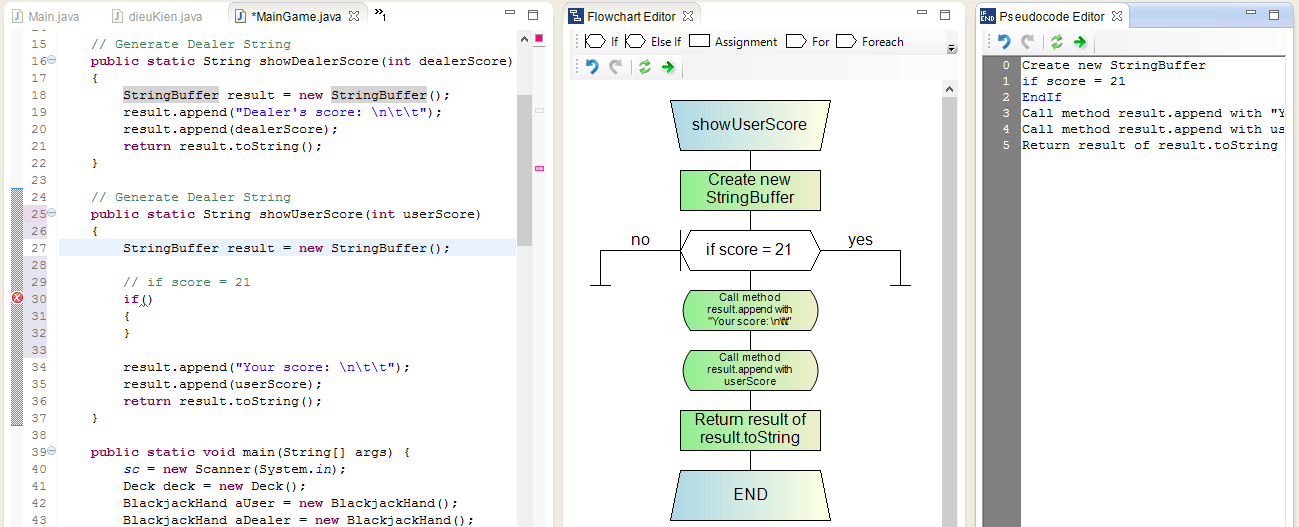
### Đồng bộ giữa mã giả, flowchart với code.

Code ban đầu khi chưa đồng bộ:



Thao tác trên trên mã giả và đồng bộ lại với code

3. Ấn commit để đồng bộ với code



2. Flow chart tự động được đồng bộ theo

1. Thao tác trên mã giả

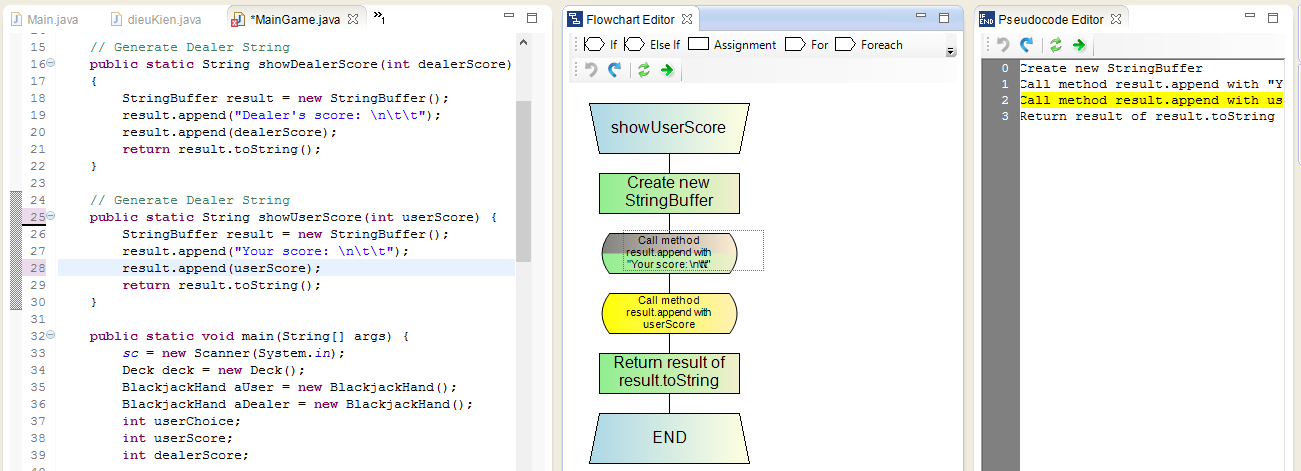
4. Code sau khi được đồng bộ

### Kéo thả giữa các thành phần có sẵn

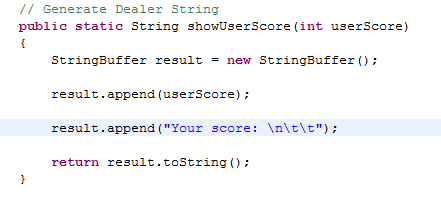
Ta kéo hàm “result.append(userScore)” lên trên đâu của hàm showUserScore

3. Ấn commit khi muốn đồng bộ với code

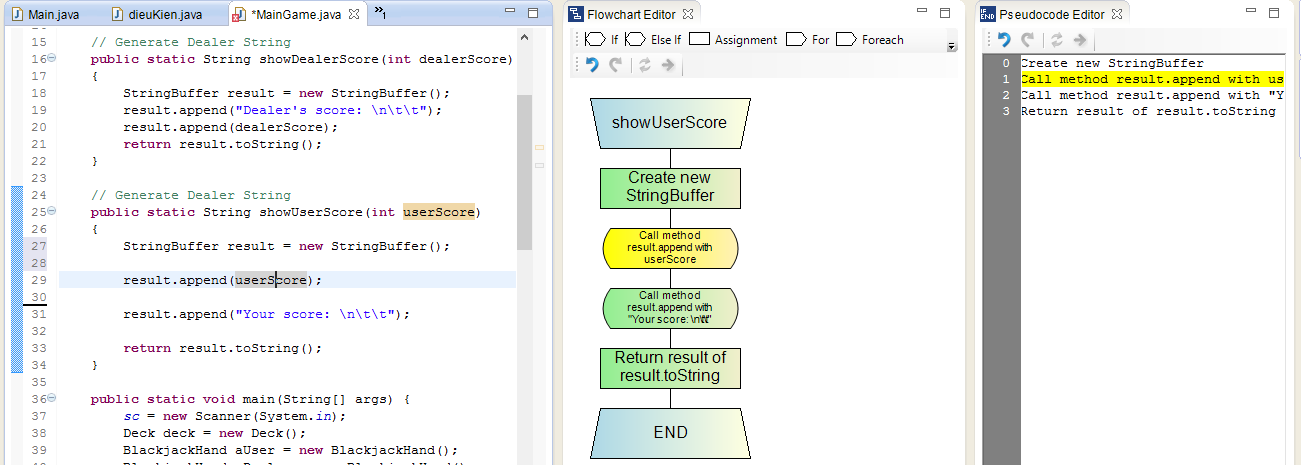
1. Click chọn



2. Kéo đến đây

Code sau khi đã đồng bộ

### Chức năng “High light” cho phép người dùng quan sát code lẫn sơ đồ rõ ràng hơn.



3. High light mã giả

2. High light flow chart

1. Click chọn code

### Xuất tài liệu

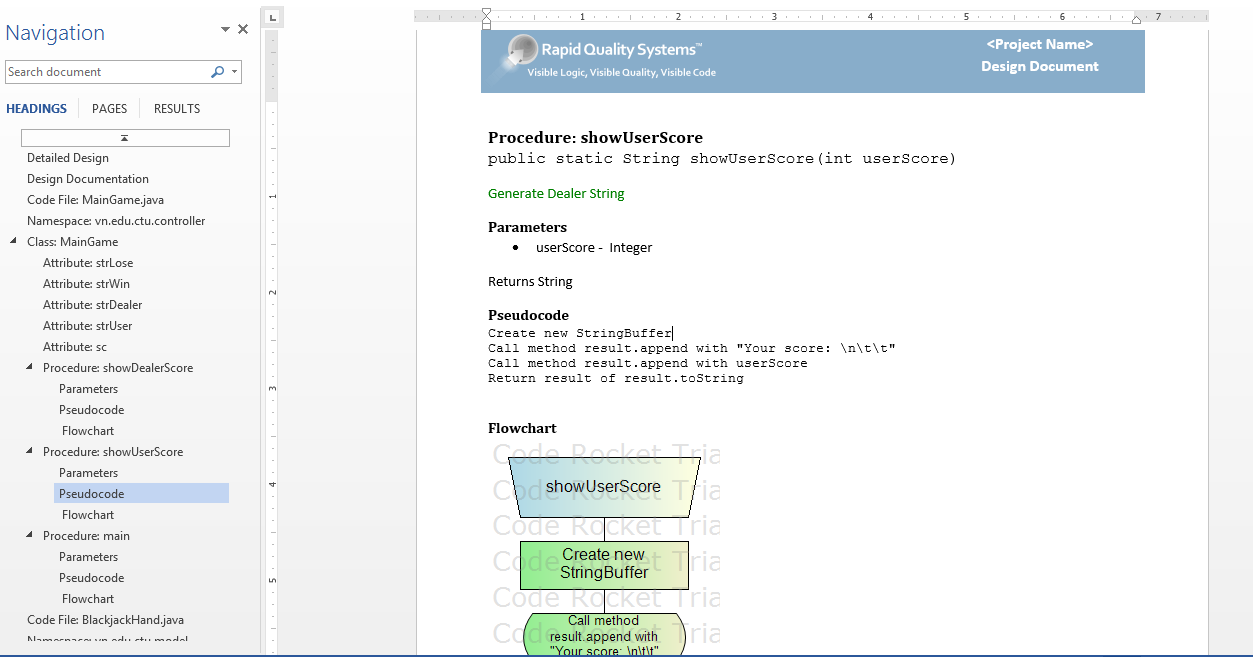
Cho phép xuất tài liệu thiết kế thành 2 dạng: HTML, Word.

Người sử dụng có thể xuất tài liệu trên phạm vi: project, package, lớp.

***Tài liệu dạng word***

Tài liệu khi xuất ra có:

* Định dạng heading -> cấu trúc trang rõ ràng, có thể sinh mục lục tự động.
* Header, footer.
* Trang bìa rõ ràng.
* Phù hợp với chuẩn tài liệu thiết kế chung thường dùng.



Tên các biến

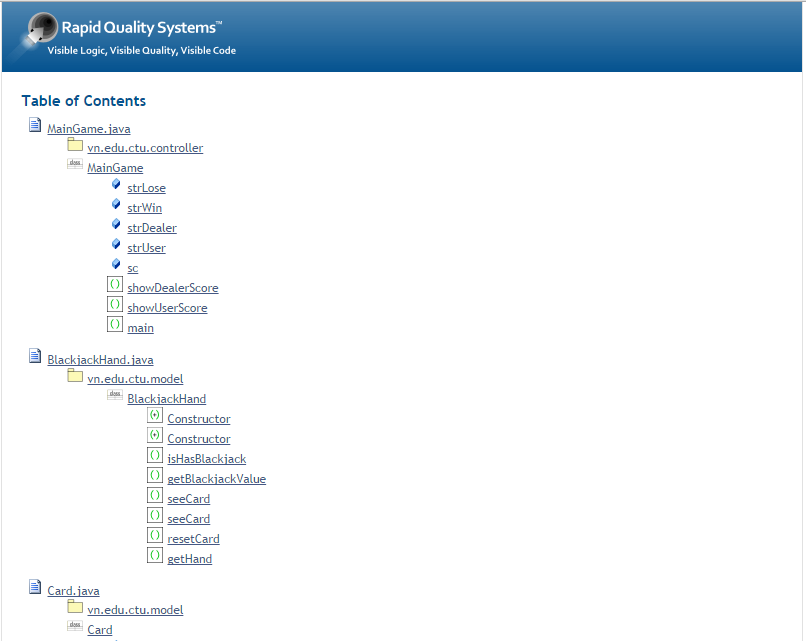
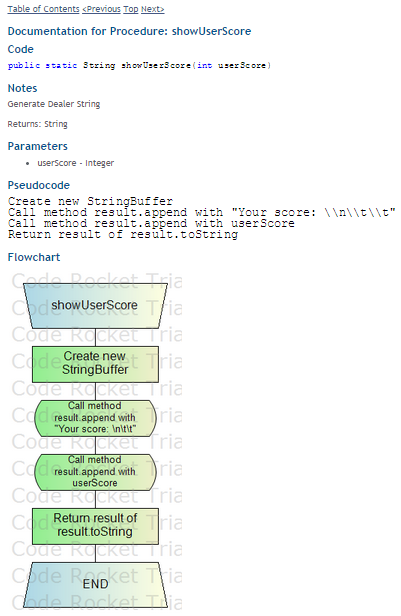
Tên phương thức hàm

Chèn các mã giả

Định dạng heading theo cấu trúc trang

***Tài liệu dạng HTML***

Phân cấp tài liệu rõ ràng, dùng các liên kết để xem các thành phần chi tiết.



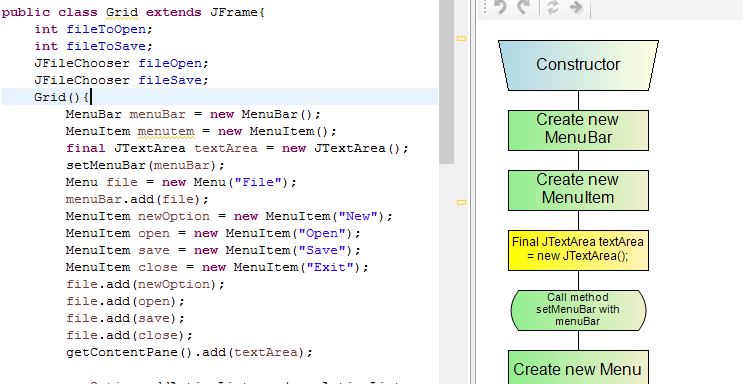
2. Kết quả

1. Click vào để xem

### Ưu và nhược điểm

Ưu điểm:

* Dựng lại bản thiết kế từ mã nguồn và sinh ra code từ bản thiết kế.
* Giảm thời gian phát triển phần mềm và chi phí.
* Nhanh chóng tạo ra các flow chart từ mã giả.
* Sự đơn giản các góc nhìn về thiết kế giúp cho mã nguồn có thể hiểu được bởi những người chưa có khái niệm về lập trình.
* Tạo “sườn code” từ các bản thiết kế.
* Comment rõ ràng.
* Đồng bộ giữa mã giả, Flow Chart với code.
* High light các thành phần được chọn bởi chuột.
* Hỗ trợ kéo thả, copy.
* “Help new team members get quickly up to speed too” - Giúp các thành viên mới dễ dàng “bảo trì và hiểu Flow chart một cách nhanh chóng” bắc kịp tiến độ.
* Sinh tài liệu ra file HTML, Word (định dạng heading cấu trúc trang).
* Khi định nghĩa phương thức xây dựng thì khi tạo ra flowchart phương thức đó mang tên là Constructor để có thể phân biệt được với các phương thức thông thường.



Nhược điểm:

* Chưa có tập hợp các lớp theo package.
* Ký hiệu của hình chưa phù hợp với các ký hiệu chung thường dùng.
* Khi định nghĩa 1 phương thức con nằm trong 1 phương thức. Thì khi sinh tài liệu, phần flowchart của phương thức con không được thể hiện. Chỉ thể hiên ở dạng gọi phương thức con.