

Devcamp

Coding Convention

Version 07.2021

Coding Convention sẽ giúp Bạn, có code chuẩn trong team.

Code dễ hiểu, debug, sửa lỗi, mở rộng, tái sử dụng (cho cá nhân và team), nâng cấp.

Refactoring: sửa code, tái cấu trúc code để sửa lỗi, nâng cấp, bàn giao người mới, làm cho code chạy nhanh hơn....Refactoring là **công việc chính** của developer (không phải coding)

Chúng ta sẽ bắt đầu **tương phản và mở rộng** trong quá trình học

Devcamp

Javascript Coding Convention Version 05.2021

Coding Convention sẽ giúp Bạn, có code chuẩn trong team.

Code dễ hiểu, debug, sửa lỗi, mở rộng, tái sử dụng (cho cá nhân và team), nâng cấp.

Refactoring: sửa code, tái cấu trúc code để sửa lỗi, nâng cấp, bàn giao người mới, làm cho code chạy nhanh hơn....Refactoring là **công việc chính** của developer (không phải coding)

Chúng ta sẽ bắt đầu **tùng phần và mở rộng** trong quá trình học

Javascript coding convention

Naming - đặt tên

Ví dụ minh họa

3

| | Đối tượng | Quy định | Ví dụ đúng | Ví dụ sai |
|-------|------------------|---|-----------------------------|-----------------------------|
| cnm10 | Function | lowerCamelCase, descriptive - có ý nghĩa | getCarByVid | getCarbyvid |
| cnm20 | Object member | camelCase | ...{ firstName: "Joe" ...} | ...{ vFirstName: "Joe" ...} |
| cnm30 | Variable (biến) | <scope>lowerCamelCase, Descriptive, có ý nghĩa | | |
| cnm40 | global | g | gSelectedMenu | vSelectedmenu |
| cnm50 | Trong function | v | vElementNewCell | elementNewCell |
| cnm60 | Tham số function | param | paramResponse | response |
| cnm70 | Trong Block | b | for (var blterator = 0;...) | for (var Iterator = 0;...) |
| cnm80 | Constant | <scope>PART1_PART2_... | gREQUEST_STATUS_OK | gRequestStatusOK |
| cnm90 | Event handler | on<elementName><Event> | onClick() | onButtonSubmitClick() |

Ngoại trừ: một số trường hợp đặc biệt ví dụ code có sẵn của library

Style - kiểu dáng

4

| | Đối tượng | Quy định | Ví dụ đúng | Ví dụ sai |
|-------|------------------|--|--|---|
| cst20 | Lùi vào | tab | Dùng tab | Dùng space |
| cst30 | operator | Có dấu cách trước và sau | bIterator < gCarDb.length | bIterator<gCarDb.length |
| cst40 | Lệnh | Mỗi lệnh trên dòng riêng. Ngắt bởi ; | | Viết 02 lệnh trên dòng; ko có dấu ; |
| cst50 | variable | Mỗi variable định nghĩa 1 dòng riêng; Phải gán giá trị ngay. | var vResult = 0; var bFound = false; | var vResult, bFound = false; |
| cst60 | function | Dùng “ use strict; ” ở dòng đầu để tránh dùng biến không khai báo; Dùng cả ở đầu của <script>... | function findCarByVid(paramCarVid) { “use strict”; ... } | function findCarByVid(paramCarVid) { var vCarFound = false; ... } |
| cst70 | Function comment | // << làm gì >> // input/start: tham số đầu vào hoặc trang thái ban đầu // output/end: kết quả trả lại, hoặc trạng thái cuối | <u>Link ví dụ minh họa</u> | |

Standard solution - giải pháp chuẩn

5

| Convention | Nội dung | Ví dụ: |
|---|--|--|
| css10 Self documenting code - code tự document | Code từng ý nhỏ (tách) để code thành dễ hiểu, dễ debug, chia để tri KHÔNG làm tắt nhiều bước cùng trên 1 dòng | <pre>ĐÚNG: vElementInputMsg = document.getElementById("inp-msg"); alert(vElementInputMsg.value); SAI: alert(document.getElementById("inp-msg").value);</pre> <p><u>Ví dụ minh họa</u></p> |
| css20 Vòng lặp for hay while | Vòng có số lần cố định: for Vòng có số lần phụ thuộc điều kiện: while | <pre>ĐÚNG: while (bIterator < gCarDb.length && !bFound) for (bIterator = 0; bIterator < gCarDb.length...) if ... Return;</pre> <p><u>Ví dụ minh họa</u></p> |
| css30 Return | Dùng tối thiểu return . Đa số chỉ nên dùng 1 lần trong 01 function . (để dễ quản trị flow code, dễ debug) | <p><u>Ví dụ minh họa</u></p> |
| css40 If-else nhiều nhánh | Tránh dùng embeeded. Hãy dùng else if | <p><u>Ví dụ minh họa</u></p> |

5

Standard solution - giải pháp chuẩn

6

| Convention | Nội dung | Ví dụ: |
|--|---|--------|
| css50 Chia để trị 03 bước - khi xử lý sự kiện trên front end | Khi dùng Javascript để xử lý sự kiện trên front-end , cần tách thành 3 bước rõ ràng. Mỗi bước là một hàm (function) riêng biệt: Link ví dụ 01) đọc dữ liệu và lưu vào một đối tượng 02) kiểm tra dữ liệu. Nếu dữ liệu hợp lệ thì thực hiện bước 03 03) xử lý dữ liệu front-end, hiển thị | |
| css60 Chia để trị 04 bước - Khi xử lý sự kiện có gọi server (API) | Khi dùng Javascript để xử lý sự kiện có tương tác với server , cần tách thành 4 bước rõ ràng. Mỗi bước là một hàm (function) riêng biệt: Link Ví dụ Javascript cơ bản Link ví dụ gọi với Jquery - Ajax 01) đọc dữ liệu và lưu vào một đối tượng 02) kiểm tra dữ liệu. Nếu hợp lệ thì thực hiện bước 03 và 04 03) gọi server, api (gửi yêu cầu request tới server) 04) xử lý dữ liệu front-end, hiển thị khi server phản hồi về | |

6

Standard solution - giải pháp chuẩn

7

| Convention | Nội dung | Ví dụ: |
|--|---|---|
| css80 Chia để trị: 04 vùng (region) trong code Javascript | <p>Cần chia code Javascript thành 04 vùng (region) để dễ quản trị hơn:</p> <pre>/** REGION 1 - Global variables - Vùng khai báo biến, hằng số, tham số TOÀN CỤC */ /** REGION 2 - Vùng gán / thực thi hàm xử lý sự kiện cho các elements */ /** REGION 3 - Event handlers - Vùng khai báo các hàm xử lý sự kiện */ /** REGION 4 - Common funtions - Vùng khai báo hàm dùng chung trong toàn bộ chương trình*/</pre> | <p>Link ví dụ</p> |
| css90 | <p>Tạo hàm onPageLoading() để xử lý việc tải trang (onload)</p> | <p>Khai báo một hàm onPageLoading(). Hàm này chứa tất cả các việc cần xử lý khi tải trang.</p> <p>Link ví dụ</p> <p>Gán hàm onPageLoading() cho sự kiện onload của thẻ body</p> |

7

Tool kiểm tra Javascript:

8

Tool kiểm tra code javascript, bước đầu dùng tạm phiên bản online của EsLint (sau sẽ dùng extension Eslint của VsCode).

Cách dùng: copy code của bạn vào để kiểm tra.

The screenshot shows the ESLint Demo - Pluggable interface. At the top, there's a navigation bar with icons for GitHub, ESLint.org, demo#eyJ0ZXh0djoiZnVuY3Rpba2gZ2VvU3VtVmVyOxJyYXkocGFyYWN1OdvW1IZXIBcnlheSkg1xuXHRch... and a user icon. Below the navigation is a search bar with placeholder 'Search the docs...' and a button labeled '2'. The main area contains a code editor with the following JavaScript code:

```
1 function getSumNumberArray(paramNumberArray) {  
2     "use strict";  
3     var vSumResult = 0;  
4     for (bI = 0; bI < paramNumberArray.length; bI++) {  
5         vSumResult += paramNumberArray[bI];  
6     }  
7     return vSumResult;  
8 }  
9  
10
```

To the right of the code editor, there's a 'Messages' panel with one error message:

4:10 - 'bI' is defined but never used. (no-unused-vars)

Below the code editor, there's a callout pointing to the '3' in the code with the text 'copy mã javascript của bạn vào đây'.

At the bottom left, there's a note: 'Want to deliver high-quality images on your websites & APPs? Try ImageKit now!' with a 'Try ImageKit now!' button.

On the right side of the interface, there are several tabs: 'Messages' (highlighted), 'Fixed Code', 'About', 'Blog', 'Demo', 'Developer guide', 'User guide', and 'Search the docs...'. A red arrow points from the text 'xử lý những vấn đề ở đây' to the 'Messages' tab.

8

Ví dụ naming:

```
//save data when editing
function saveDataInEditMode() {
    "use strict";
    const INVALID_DATA = 10; 2
    try {
        //validate data
        if ($("#voucherCodeInput").val().trim() === "" || $("#discountInput").val().trim() === "") {
            throw(INVALID_DATA);
        }
        //update in array
        //debugger;
        var WoucherBeingUpdated = findVoucherById(gvoucherBeingEdited.id);
        WoucherBeingUpdated.voucherCode = $("#voucherCodeInput").val().trim();
        WoucherBeingUpdated.discount = parseInt($("#discountInput").val().trim());
        //update in table
        g$RowBeingEdited.children("td:eq('' + gCOL_VOUCHER_CODE + '')").html($("#voucherCodeInput").val().trim()); // cũ eq('1')
        g$RowBeingEdited.children("td:eq('' + gCOL_DISCOUNT + '')").html($("#discountInput").val().trim()); // cũ eq('2')
4
        //change form mode
        gFormEditMode = gNODE_INSERT; 5
        //message update thành công
        alert ("update thành công");
    }
    catch (err) { một số ngoại lệ
        if (err === INVALID_DATA) {
            alert("invalid data: empty value inputted");
        }
        else
            throw(err);
    }
}
```

Có thể có 1 số ít ngoại lệ

Ví dụ: Self documenting code - dài hơn, nhưng dễ hiểu, dễ debug, dễ check value để tìm vết

```
68 | personInput.firstName = document.getElementById("firstNameInput").value;
69 | //get firstName
70 | var firstNameInputElement = document.getElementById("firstNameInput");
71 | personInput.firstName = firstNameInputElement.value;
72 |
73 | //check firstName
74 | console.log(firstName); //check firstName
75 |
76 | //get firstName
77 | var firstNameInputElement = document.getElementById("firstNameInput");
78 | console.log(firstNameInputElement); //check firstNameInputElement
79 | personInput.firstName = firstNameInputElement.value;
80 | console.log(firstName); //check firstName
```

Dài hơn, nhưng dễ
hiểu hơn

Có nhiều điểm để
check, hay debug
hơn.
Ở ví dụ bên dưới,
bạn dễ dàng phát
hiện ra vấn đề, nếu
bạn ko truy xuất
element vì lỗi đặt
tên, hoặc viết sai tên
hàm getElementById

10

Javascript coding convention

Ví dụ: sử dụng vòng lặp

| Ví dụ | Pattern áp dụng & best practices | Ghi chú/Ví dụ |
|---|----------------------------------|---|
| Vòng lặp: số vòng cố định | for | In hết các attributes của 01 DOM elements; Tim số lớn nhất |
| Vòng lặp: đến 1 condition nào đó còn đúng - số vòng không cố định | while | Tim person trong array sử dụng citizenID (điều kiện - chưa tìm thấy thì còn chạy) <pre>var voucherFound = false; var iterator = 0; while (!voucherFound && iterator < voucherDb.length) { if (voucherDb[iterator].voucherId === inputVoucherId) { voucherFound = true; } else { iterator++; } }</pre> |

11

Javascript coding convention

Ví dụ: **While** vòng lặp, loop đến lúc điều kiện còn đúng

- 1) Tìm đến khi chưa tìm thấy và index còn trong range
- 2)Ở đây đã có self documenting.

```
36 //search a car in carDB by a VID as input
37 // return null if not found
38 function getCarByVID(carVID) {
39     var carFound = false;
40     var carIndex = 0;
41     // search until found
42     while (!carFound && carIndex < carDB.length ) {
43         if ( carDB[carIndex].vid == carVID) {
44             carFound = true;
45         }
46         else {
47             carIndex++;
48         }
49     } // return a car if found, or return null
50     if (carFound) {
51         return carDB[carIndex];
52     }
53     else {
54         return null;
55     }
56 }
57 }
```

Ví dụ: **sử dụng return 01 lần duy nhất** (trừ code kiểm tra tham số của function).

Để kế hoạch, các bước rõ và thẳng một đường, từ đầu tới cuối..

Dùng nhiều return, nhiều phân nhánh, khó debug, dễ bị ròi.

```

36 //search a car in carDB by a VID as input
37 // return null if not found
38 function getCarByVID(carVID) {
39     var carFound = false;
40     var carIndex = 0;
41     // search until found
42     while ( !carFound && carIndex < carDB.length ) {
43         if ( carDB[carIndex].vid == carVID ) {
44             carFound = true;
45         } else {
46             carIndex++;
47         }
48     }
49     // return a car if found, or return null
50     if (carFound) {
51         return carDB[carIndex];
52     }
53     else {
54         return null;
55     }
56 }
57 
```

```

36 //search a car in carDB by a VID as input
37 // return null if not found
38 function getCarByVID(carVID) {
39     var carFound = false;
40     var carIndex = 0;
41     // search until found
42     while ( !carFound && carIndex < carDB.length ) {
43         if ( carDB[carIndex].vid == carVID ) {
44             carFound = true;
45         } else {
46             carIndex++;
47         }
48     }
49     // return a car if found, or return null
50     if (carFound) {
51         var returnCar = null;
52         if (carFound) {
53             returnCar = carDB[carIndex];
54         }
55     }
56     return returnCar;
57 }

01 return
13 
```

Ví dụ:

tránh dùng if-else lồng (embedded),

Hãy dùng

else if

```
// function ban đầu, các lệnh if else bị lồng vào nhau
function myFunction () {
    var vGreeting = "no greeting";
    var vTime = new Date() .getHours () ; //1 ấy thời gian trong ngày
    if (vTime < 10) {
        vGreeting = "Good morning";
    }
    else {
        if (vTime < 20) {
            vGreeting = "Good day";
        }
        else {
            vGreeting = "Good evening";
        }
    }
    document.getElementById ("demo") .innerHTML = vGreeting;
}

// đã refactor thành các nhánh rõ ràng
function myFunctionRefactored () {
    var vGreeting = "no greeting";
    var vTime = new Date () .getHours () ; //1 ấy thời gian trong ngày
    if (vTime < 10) {
        vGreeting = "Good morning";
    } else if (vTime < 20) {
        vGreeting = "Good day";
    } else {
        vGreeting = "Good evening";
    }
}
document.getElementById ("demo-refactored") .innerHTML = vGreeting;
```

Ví dụ Standard solution - giải pháp chuẩn - chia để trị ở các nơi cần xử lý dữ liệu và hiện thị - chỉ front-end

15

```
85     function onButtonSaveDataClick() {  
86         "use strict";  
87         var vUserDataObject = [  
88             firstName : "",  
89             lastName : "",  
90             age: 0,  
91             email: ""  
92         ];  
93         //1. đọc dữ liệu  
94         readInputData(vUserDataObject);  
95         //2 check dữ liệu,  
96         var vValidateData = checkSubmitData(vUserDataObject);  
97         if (vValidateData) {  
98             //3 xu ly hien thi front-end: neu du lieu ok, add 01 row vao bang  
99             addOneUserToTable(vUserDataObject); //hàng này chưa code  
100        }  
101    }  
102
```

Ko gọi server,
chỉ xử lý tại
front-end

15

Ví dụ Standard solution - giải pháp chuẩn - chia để trị ở các nơi cần xử lý dữ liệu và hiện thị

```
function onVoucherCheckClick() {  
    "use strict";  
    //cau truc du lieu se lam viec  
    var vObjectVoucher = {  
        "voucherCode": ""  
    };  
    //1. lay gia tri nhap tren form  
    readVoucherData(vObjectVoucher);  
    //2. validation chua nay co the co validation  
    var validateResult = validateVoucherData(vObjectVoucher);  
    if (validateResult) {  
        //3. call rest api  
        var vXmlHttpRequest = new XMLHttpRequest();  
        callRestApiCheckVoucher(vObjectVoucher, vXmlHttpRequest);  
        //4. process response  
        vXmlHttpRequest.onreadystatechange = function() {  
            if (vXmlHttpRequest.readyState == REQUEST_STATE_DONE) {  
                processResponse(vXmlHttpRequest);  
            }  
        }  
    }  
}
```

```
// update user info
function callRestApiUpdate() {
  "use strict";
  // B1: Get data
  //dates to be sent
  var objReqUpdateData = {
    firstname: "Linh", //ban có thể sửa các giá trị này để thử, và lại get lại data
    lastname: "Do",
    subject: "On business 300",
    country: "VN"
  };
  // B2: validate
  var validateData = validateData(objReqUpdateData);
  // B3 xu ly nghiệp vụ
  if(validateData) {
    var XMLHttpRequest = new XMLHttpRequest(); // new Httprequest instance
    var XMLHttpRequest = new XMLHttpRequest();
    XMLHttpRequest.onreadystatechange = function() {
      if(XMLHttpRequest.readyState == XMLHttpRequest.READY_STATE_FINISH_OK && XMLHttpRequest.status == XMLHttpRequest.STATUS_OK) {
        responseHandlerXML(XMLHttpRequest);
      }
    }
    XMLHttpRequest.open("PUT", "/api/v1/users");
    XMLHttpRequest.setRequestHeader("Content-Type", "application/json");
    XMLHttpRequest.send(JSON.stringify(validateData));
  }
}
```

Gọi server javascript cơ bản

Ví dụ Standard solution - giải pháp chuẩn - chia để trị ở các nơi cần xử lý dữ liệu và hiện thị - gọi server có Ajax Jquery

17

```
138 //update user info
139 function callAjaxRestApiUpdate() {
140   "use strict";
141   console.log("call Ajax Rest API Update User")
142   // B1: get data
143   // dùng guserId
144   //data to be sent
145   var vObjectRequestData = getRequestData();
146
147   // B2: validate data
148   var vIsValidate = validateData(vObjectRequestData);
149
150   if(vIsValidate) {
151     // B3: process update user
152     $.ajax({
153       url: "http://42.115.221.44:8080/devcamp-register-java-api/users/" + guserId,
154       type: 'PUT',
155       dataType: 'json', // added data type
156       contentType: "application/json; charset=UTF-8",
157       data: JSON.stringify(vObjectRequestData),
158       success: function(res) {
159         // B4: handle front-end
160         handleFrontEndAjaxCallUpdateUser(res); ④
161       },
162       error: function (ajaxContext) {
163         alert(ajaxContext.responseText)
164       }
165     });
166
167 }
```

17

Style - kiểu dáng - comment cho function

```
//đỗ dữ liệu từ response vào bảng, hiện các button chi tiết
//input/start:
//    - paramXmlHttpUser: request đã dùng @i server, và có responsetext chứa dữ liệu
//output/end:
//    - dữ liệu hiện tại trên bảng;
//    - các button chi tiết được hiện và có sẵn dữ liệu dataset, có event handler
function displayListView(paramXmlHttpUser) {
    "use strict";
    var vUserListObject;
    var vJSONText = paramXmlHttpUser.responsetext;
    vUserListObject = JSON.parse(vJSONText);
    console.log("length là: " + vUserListObject.length);
    //load data to html table
    showDataOnTable(vUserListObject);
    //add event listener to chi tiet buttons
    addListenerEventInfoButton();
}
```

CRUD - standard solution

```
// Các biến toàn cục hằng số Form mode: 4
var gFORM_MODE_NORMAL = "Normal";
var gFORM_MODE_INSERT = "Insert";
var gFORM_MODE_UPDATE = "Update";
var gFORM_MODE_DELETE = "Delete";

// Biến toàn cục cho trạng thái của form:
// mặc định ban đầu là trạng thái Normal
var gFormMode = gFORM_MODE_NORMAL;

// Khi thực hiện xong sự kiện, luôn
// reset form về trạng thái ban đầu
Normal
```

```
function onBttnSaveVoucherClick() {
    // B0: khai báo đối tượng voucher để chứa data
    var vVoucherData = {
        "id": 0,
        "voucherCode": "",
        "discount": -1
    };

    // B1: Thu thập data
    getVoucherData(vVoucherData);
    // B2: Validate data
    var vIsDataValidate = validateData(vVoucherData);

    if (vIsDataValidate) {
        // B3: thực hiện nghiệp vụ lưu trữ (thêm / sửa) data
        saveVoucher(vVoucherData);
        // B4: Cập nhật lại giao diện, thông báo cho người dùng
        // và reset form về trạng thái ban đầu
        alert("Cập nhật dữ liệu thành công!");
        loadDataToVoucherTable(gVoucherObjects);
        resetFormToStart();
    }
}
```

bên dưới

// Dù thực hiện sự kiện nào cũng cần thực hiện cú 5 bước

04 Regions trong code Javascript

```
/* * * REGION 1 - Global variables - Vùng khai báo biến, hằng số, tham số TOÀN CỤC */
var gFORM_MODE_INSERT = "Insert";
var gFormMode = gFORM_MODE_INSERT;

/* * * REGION 2 - Vùng gán / thay đổi thi hàm xử lý sự kiện cho các elements */
// gán hàm xử lý sự kiện change cho select Country
selectCountryElement.addEventListener ('change' , onSelectElementChange );
// thêm sự kiện click cho nút save data
$( "#btn-save-data" ).on ("click" , onSaveSaveDataClick );

/* * * REGION 3 - Event handlers - Vùng khai báo các hàm xử lý sự kiện */
function onSaveSaveDataClick () {
    var vVoucherData = {};
    getVoucherData (vVoucherData );
}

// hàm thực hiện khi load trang
function onPageLoading () {
}

/* * * REGION 4 - Common functions - Vùng khai báo hàm dùng chung trong toàn bộ chương trình*/
// Hàm thực hiện việc insert voucher
function insertVoucher () {
}

// hàm thu thập dữ liệu voucher
function getVoucherData (paramVoucherData ) {
}
```

Tạo và gán hàm onPageLoading() để thực thi sự kiện tải trang onload

```
14 </head>
15 <!-- Gán hàm xử lý sự kiện tải trang (onload) ở thẻ body -->
16 <body onload="onPageLoading()">
17   <div class="container-fluid">
18     <div class="row">...
40   </div>
41 </div>
42 </body>
43 <script>
44   "use strict";
45   /*** REGION 1 - Global variables - Vùng khai báo biến, hàng số, tham số TOÀN CỤC */
46
47   /*** REGION 2 - Vùng gán / thực thi sự kiện cho các elements */
48
49   /*** REGION 3 - Event handlers - Vùng khai báo các hàm xử lý sự kiện */
50   // Hàm thực hiện khi load trang
51   function onPageLoading() {
52     getAllOrders();
53   }
54   /*** REGION 4 - Common functions - Vùng khai báo hàm dùng chung trong toàn bộ chương trình*/
55   // Hàm thực hiện việc load all orders từ server và load vào table
56   >   function getAllOrders() { ...
73 </script>
74 </html>
```

Devcamp

HTML Coding Convention

Version 05.2021

Coding Convention sẽ giúp Bạn, có code chuẩn trong team.

Code dễ hiểu, debug, sửa lỗi, mở rộng, tái sử dụng (cho cá nhân và team), nâng cấp.

Refactoring: sửa code, tái cấu trúc code để sửa lỗi, nâng cấp, bàn giao người mới, làm cho code chạy nhanh hơn....Refactoring là **công việc chính** của developer (không phải coding)

Chúng ta sẽ bắt đầu **từng phần và mở rộng** trong quá trình học

HTML - 03 thành phần chính - 03 components (chú ý: sẽ mở rộng trong quá trình học)

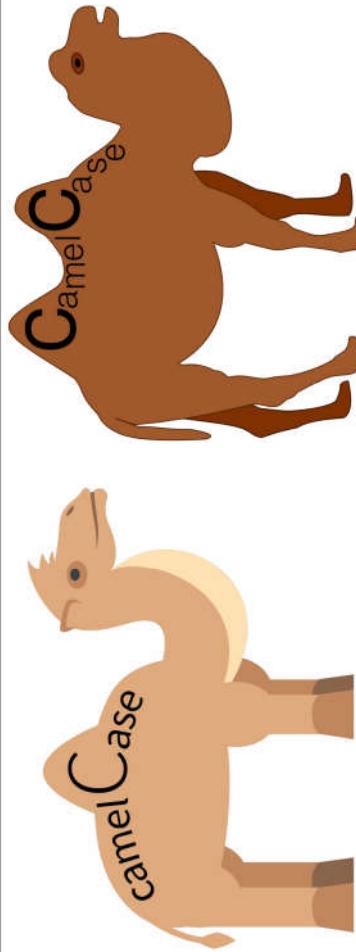
- Naming:
 - [Hnm10] Dùng element name lowercase (<p> thay cho <P>)
 - [Hnm20] id của element <**type**>-<**part1**>-<**part2**> **btn-save-data** **input-first-name**
- Style - hình thức code
 - [hst10] Chỉ dùng line trống (blank line) để tách block lớn của code, không để line trống vô cớ
 - [hst20] Comment block của code
 - [hst30] Element phải đóng (</div>, </form>, </p>...)
- Standard solution/pattern: các giải pháp chuẩn - hay pattern mà team thống nhất cho 1 số vấn đề cụ thể có thể về code hoặc thiết kế code.
 - [hss10] Có <title>
 - [hss20] Có <header>
 - [hss30] Dùng External css cho css dài

HTML coding convention

Naming - đặt tên

24

| | Đối tượng | Quy định | Ví dụ đúng | Ví dụ sai |
|-------|-----------------|---|--------------------------|----------------------------|
| jnm10 | Phương thức | camelCase, descriptive - có ý nghĩa | getCarByVid | getCarbyvid |
| jnm20 | Thuộc tính | camelCase, private | private String firstName | public String firstName |
| jnm30 | Variable (biến) | camelCase, Descriptive - có ý nghĩa | variableName | VariableName, variablename |
| jnm40 | Class, Model | CamelCase, Descriptive - có ý nghĩa | ClassName | className, classname |
| jnm50 | Constant | Phải khai báo final PART1 PART2 ... | REQUEST_STATUS_OK | RequestStatusOK |



Nguyên tắc: một số trường hợp đặc biệt ví dụ code có sẵn của library

Style - kiểu dáng

25

| | Đối tượng | Quy định | Ví dụ đúng | Ví dụ sai |
|------|---------------------------------|--|---|---|
| js10 | comment | // 1 - 3 dòng /* / block > 3 dòng | //check the input value | Rất nhiều dòng nhưng dùng // trên từng dòng |
| js20 | Lùi vào | Tab, Hai dòng code cách nhau một bậc thì sẽ cách nhau một đơn vị thụt đầu dòng. | Dùng tab | Dùng space |
| js30 | operator | Có dấu cách trước và sau | count += 1; | count+=1; |
| js40 | Lệnh | Mỗi lệnh trên dòng riêng. Ngắt bởi ; | | Viết 02 lệnh trên dòng; |
| js50 | variable | Mỗi variable định nghĩa 1 dòng riêng; Phải gán giá trị ngay. | int result = 0; Int count = 0; boolean found = false; | |
| js60 | method comment | /** * << làm gì >> * * @param input/start tham số đầu vào hoặc trạng thái ban đầu * @return output/end kết quả trả lại, hoặc trạng thái cuối */ | | |

Standard solution - giải pháp chuẩn

26

| Convention | Nội dung | Ví dụ: |
|--|---|--------|
| jss10 Self documenting code - code tự document | Code từng ý nhỏ (tách) để code thành dễ hiểu, dễ debug . KHÔNG làm tắt nhiều bước cùng trên 1 dòng. Ví dụ thêm những biến mới có ý nghĩa, giúp ta debug tối hơn, code dễ hiểu hơn. | |
| jss20 Vòng lặp for hay while | Vòng có số lần cố định: for Vòng có số lần phụ thuộc điều kiện: while | |
| jss30 Return | Dùng tối thiểu return . Đa số chỉ nên dùng 1 lần trong 01 function . (để dễ quản trị flowcode, dễ debug) | |
| jss40 Phương thức trong model Java | Phương thức trong model chỉ sử dụng những thuộc tính có trong model đó và phương thức đó phải có ý nghĩa đối với model đó | |

26

Standard solution - giải pháp chuẩn

27

| | Convention | Nội dung | Ví dụ: |
|-------|---|--|--------|
| jss50 | Tổ chức thư mục (package) | Các class có cùng ý nghĩa cần để chung vào 1 package. | |
| jss60 | Các phần cơ bản bắt buộc có của một model | <ul style="list-style-type: none">- Thuộc tính- Khởi tạo (không tham số, có tham số)- Các phương thức Get, Set | |

27

Format đặt tên RestAPI (Ví dụ CRUD user-order)

a. GetMapping

- `@GetMapping("/users")` : Get ALL Users (Lấy danh sách toàn bộ Users)
- `@GetMapping("/{userid}")` : Get Info Users By ID (Lấy thông tin User bởi userID)
- `@GetMapping("/{userid}/orders")` : Get ALL Orders Of User (Lấy danh sách Orders của một User)
- `@GetMapping("/{userid}/orders/{orderid}")` : Get Info Orders Of User By ID (Lấy thông tin Order của một User bởi orderID)
-

Format đặt tên RestAPI (Ví dụ CRUD user-order)

b. PostMapping

- `@PostMapping("/users")` : Create Users (Tạo Users)
- `@PostMapping("/users/{userid}/orders")` : Create order of user (Tạo order cho user)

c. PutMapping

- `@PutMapping("/users/{userid}")` : Update User Info (Cập nhật thông tin User)
- `@PutMapping("/users/{userid}/orders/{orderid}")` : Update Order Info of User (Cập nhật thông tin Order của User)

c. DeleteMapping

- `@DeleteMapping("/users/{userid}")` : Xóa thông tin user
- `@DeleteMapping("/users/{userid}/orders/{orderid}")` : Xóa thông tin Order của User

Standard solution: các bước tạo Entity

| Số | Action | Ví dụ |
|-----|--|--|
| 1 | Tạo Entity, khai báo table | <pre>@Entity @Table(name = "country") public class CCountry {}</pre> |
| 2 | Khai báo ID (column PK) | <pre>@Id @GeneratedValue(strategy = GenerationType.IDENTITY) private Long id;</pre> |
| 3 | Khai báo các thuộc tính (column cho table) | <pre>@Column(name = "ma_voucher", unique = true) private String maVoucher;</pre> |
| 3.1 | Khai báo validate | <pre>@NotNull(message = "Phải có mã giảm giá")//link tham khảo @Size(min = 3, message = "Mã voucher phải có ít nhất 3 ký tự")</pre> |
| 4 | Khai báo quan hệ 1-n | <pre>@OneToMany(targetEntity = CRegion.class, cascade = CascadeType.ALL) @JoinColumn(name = "country_id") private List<CRegion> regions; @ManyToOne @JsonIgnore private CCountry country;</pre> <pre>@OneToOne(fetch = FetchType.LAZY, mappedBy = "createdBy") private Set<Post> posts; @ManyToOne(fetch = FetchType.LAZY) @JoinColumn(name = "created_by", nullable = false) private User createdBy;</pre> |

Standard solution: Các bước tạo Entity

| Số | Action | Ví dụ |
|----|----------------------------------|--|
| 5 | Khai báo quan hệ 1-1 | <pre>@OneToOne(fetch = FetchType.LAZY, cascade = CascadeType.ALL, mappedBy = "user") private Profile profile; @OneToOne(fetch = FetchType.LAZY, optional = false) @JoinColumn(name = "user_id", nullable = false) private User user;</pre> |
| 6 | Khai báo quan hệ n-n | <pre>@ManyToMany(fetch = FetchType.LAZY, cascade = { CascadeType.PERSIST, CascadeType.MERGE }) @JoinTable(name = "post_tags", joinColumns = { @JoinColumn(name = "post_id") }, inverseJoinColumns = { @JoinColumn(name = "tag_id") }) private Set<HashTag> tags; @ManyToMany(fetch = FetchType.LAZY, cascade = { CascadeType.PERSIST, CascadeType.MERGE }, mappedBy = "tags") private Set<Post> posts;</pre> |
| 7 | Tạo get, set, constructor method | |

Standard solution: Các bước tạo Repository

| Số | Action | Ví dụ |
|----|--|---|
| 1 | Tạo interface kế thừa JpaRepository | public interface IVoucherRepository extends JpaRepository<CVoucher, Long> {} |
| 2 | Tạo các method theo pattern để lấy dữ liệu | List<CRegion> findByCountryId(Long countryId); <u>//tham khảo link</u> Optional<CRegion> findByIdAndCountryId(Long id, Long instructord); |
| 3 | Tạo method cho JPA query | |

Standard solution: Các bước tạo Controller

| Sđt | Action | Ví dụ |
|-----|---|---|
| 1 | Tạo Controller và khai báo | <pre>@RestController public class CountryController {}</pre> |
| 2 | Khai báo các repository cần dùng | <pre>@Autowired private CountryRepository countryRepository;</pre> |
| 3 | Tạo REST API | <pre>@CrossOrigin</pre> |
| 3.1 | Khai báo phương thức và tên API cần tạo | <pre>@PostMapping("/country/create")</pre> |
| 3.2 | Khai báo method ngay phía dưới chú ý dữ liệu trả về, tên hàm (có ý nghĩa) và các tham số (nếu có) | <pre>public ResponseEntity<Object> updateCountry(@PathVariable("id") Long id, @RequestBody CCountry cCountry)</pre> |
| 3.3 | Tạo một khối try catch trong hàm để xử lý khi code lỗi | <pre>try {...} catch (Exception e) {...}</pre> |
| 3.4 | Dùng repository để lấy dữ liệu từ CSDL (sử dụng tham số truyền vào) | <pre>countryRepository.findById(id); countryRepository.deleteById(id);</pre> |
| 3.5 | Xử lý, validate dữ liệu mà repository trả về | <pre>countryRepository.findById(id).isPresent()</pre> |
| 3.6 | Return ResponseEntity (Chú ý kiểu dữ liệu của hàm và http response status) | <pre>return new ResponseEntity<>(savedRole, HttpStatus.CREATED);</pre> |