**Naming Guideline**

**Hello World!**

* 1. 공통 규칙  
     A. 파스칼 표기법(PascalCase)과 카멜 표기법(camelCase)를 사용한다.
     1. – 파스칼 표기법 : 모든 단어에서 첫번째 문자는 대문자이며 나머지는 소문자이다.(예: PascalCase)  
        – 카멜 표기법 : 최초에 사용된 단어를 제외한 첫번째 문자가 대문자이며 나머지는 소문자이다.(예: camelCase)
     2. 명칭에 약어는 최대한 사용하지않도록 한다.(단, 일반적으로 사용되는 약어는 그대로 사용한다)
     3. 반의어는 반드시 대응하는 개념으로 사용해야 한다.

– get/set  
– add/remove  
– create/destroy  
– start/stop  
– insert/delete  
– increment/decrement  
– old/new  
– begin/end  
– first/last  
– up/down  
– min/max  
– next/previous  
– open/close  
– show/hide  
– suspend/resume  
– parent/child

1. 패키지(Package) 명명 규칙
   * 1. 패키지명은 표준 패턴을 따라야 한다.  
        예) [com].[회사명].[프로젝트명].[최상위패키지명].[하위패키지명]…
     2. 패키지명으로는 가급적 한 단어의 명사를 사용한다.  
        – 좋은 예 ::  
          com.nexchal.djinn.member.object  
        – 나쁜 예 ::  
          djinn.memberObject
2. 클래스(Class) 명명 규칙
   * 1. 클래스명에는 파스칼 표기법을 사용한다.  
        예) public class HelloWorld {}
     2. 인터페이스에는 특별한 접두사나 접미사를 사용하지않고 파스칼 표기법을 사용한다.  
        예) public interface Animal {}
     3. 인터페이스를 구현한 클래스에는 특별한 접두사나 접미사를 사용하지않고 파스칼 표기법을 사용한다.  
        예) public class Tiger implements Animal {}
     4. 추상클래스에는 특별한 접두사나 접미사를 사용하지않고 파스칼 표기법을 사용한다.  
        예) public abstract class Animal {}
3. 메소드(Method) 명명 규칙
   * 1. 메소드명에는 파스칼 표기법을 사용한다.  
        예) public void SayHello(string name) {}
     2. 속성(attribute)에 접근하는 메소드명의 접두사는 ‘get’, ‘set’를 사용한다.

예) public class Tiger impliments Animal {  
   private String color = null;

public void setColor(String color) {  
     this.color = color;  
   }

public String getColor() {  
     return this.color;  
   }

* + 1. 데이터를 조회하는 메소드명의 접두사는 ‘find’을 사용한다.  
       예) public String findMember(String memberId) {}
    2. 데이터를 입력하는 메소드명의 접두사는 ‘input’을 사용한다.  
       예) public int inputMember(HashMap member) {}
    3. 데이터를 변경하는 메소드명의 접두사는 ‘modify’을 사용한다.  
       예) public int modifyMember(HashMap member) {}
    4. 데이터를 삭제하는 메소드명의 접두사는 ‘delete’을 사용한다.  
       예) public int deleteMember(String memberId) {}
    5. 초기화하는 메소드명의 접두사는 ‘init’을 사용한다.  
       예) public String initPassword(String memberId) {}
    6. 반환값의 타입이 boolean인 메소드는 접두사로 ‘is’를 사용한다.  
       예) public boolean isMember(String memberId) {}

1. 변수(Variable) 명명 규칙
   * 1. 변수와 메소드의 파라미터에는 카멜 표기법을 사용한다.  
        예) private int totalCount = 0;

public void SayHello(String name)  
{  
  String fullMessage = “Hello ” + name;  
  …  
}

* + 1. 변수명에 헝가리안 표기법을 사용하지 않는다.  
       – 헝가리안 표기법(Hungarian Notation) : 변수명을 지을때 한 단어 혹은 두 단어정도로 된 접두사를 사용하여 변수의 데이터 타입이나 자료형등의 성질이나 특성을 나타내는 방법이다.(예 : nHeight, strName, bStopYn…)  
       – 예전에는 많이 사용되었으나 현재는 추천하지 않는 편이다.
    2. 변수에 모든 의미를 충분히 담는다.(약어의 사용을 자제)  
       – 좋은 예 ::  
         String address  
         int salary

– 나쁜 예 ::  
  String nam  
  String addr  
  int sal

* + 1. 한 글자로 된 이름을 사용하지 않는다.
    2. 반복문에서 인덱스로 사용할 변수는 i, j, k… 등으로 사용한다.
    3. 지역변수와 멤버변수(전역변수)는 변수명 앞에 밑줄(\_)을 사용하여 구별한다.(멤버변수에 사용)
    4. 프로그래밍언어에서 미리 지정한 키워드와 비슷한 이름은 사용하지 않는다.
    5. boolean 타입의 변수는 접두사로 ‘is’를 사용한다.
    6. 구성요소를 나타내는 변수는 해당 요소 타입 이름을 접미어로 사용한다.
    7. UI 구성요소에 적당한 접두사를 사용한다.

1. 그 외의 규칙
   * 1. 파일명은 파스칼 표기법을 사용한다.

**적용**

**Java > Service의 Class 와 Method의 Naming은 Transaction에 영향을 주기 때문에 Naming Rule을 엄수해야 합니다.**

* [/WEB-INF/confgi/spring/spring-mybatis.xml] 의transactionManager부분 참조.

**Package**

* 소문자로만 이루어진 최소 단위의 명사형 단어 (부득이 단어 조합이 될 경우에도 소문자만을 이용하여 단어 조합)

Ex) member / casestudy / boardtype

**Class**

대문자로 시작하는 CamelCase Style의 명사형 단어에 접미어 조합.

* MemeberController / CaseStudyService / BoardTypeDao / QnaServiceImpl ```

**각 Layer 별 접미어**

* **Model** : 접미어 없음
* **Controller** : *\*Controller*
* **Service Interface** : *\*Service*
* **Service Implement** : *\*ServiceImpl*
* **Dao Interface** : *\*Dao*

//Example Member

net.okjsp.member.MemberController

net.okjsp.member.model.Member

net.okjsp.member.service.MemberService

net.okjsp.member.service.MemberServiceImpl

net.okjsp.member.dao.MemberDao

//Example CaseStudy

net.okjsp.casestudy.CaseStudyController

net.okjsp.casestudy.model.CaseStudy

net.okjsp.casestudy.service.CaseStudyService

net.okjsp.casestudy.service.CaseStudyServiceImpl

net.okjsp.casestudy.dao.CaseStudyDao

**Method**

소문자로 시작하는 cameCase Style의 동사+명사의 조합.

* getMemberInfo / createMember / insertMember / updateCaseStudyStatus ```

**기능별 접두어 정의**

**Service**

* **Create** : *create\** 또는 *add\**
* **Retrieve** : *get\** 또는 *check\**
* **Update** : *modify\** 또는 *change\**
* **Destroy** : *destroy\** 또는 *remove\**

//Example MemberService

public List<Member> getList();

public Member getOne(int id):

public boolean checkDuplicate(String loginId);

public boolean create();

public boolean modify(int id);

public boolean destroy(int id);

public boolean addAuth(int id);

public boolean removeAuth(int id);

**Dao**

* **Insert** : *insert\**
* **Select** : *select\**
* **Update** : *update\**
* **Delete** : *delete\**

//Example MemberDao

public List<Member> selectList();

public Member selectOne(int id):

public int selectCount(String loginId);

public boolean insert();

public boolean update(int id);

public boolean updateStatus(int id);

public boolean delete(int id);

public boolean insertAuth(int id);

public boolean deleteAuth(int id);

* **JSP \* File 과 Folder 모두 동일하게 소문자 Under score(\_) 구분의 명사 조합.**
* **Folder 는 단위 프로그램 기준으로 최소단위의 단어 사용.**
* **File은 File 명 만으로도 해당 프로그램을 알 수 있게 `프로그램명\_액션명.jsp` 조합으로 작성.**

/member/member\_list.jsp

/member/member\_form.jsp

/case\_study/case\_study\_list.jsp

/case\_study/case\_study\_result.jsp

**Module Name**

Module 및 하위 프로그램 명은 대문자로 시작하는 CamelCase Style의 명사형 단어에 접미어 조합.

* **Module** : \*App
* **Factory** : 접미어 없음
* **Controller** : \*+액션명+Ctrl
* **Provider** : \*Provider
* **Service** : \*Service

// Module

var sampleApp = angular.module('SampleApp', ...);

// Factory

sampleApp.factory('Sample', function(){...});

// Provider

sampleApp.provider('SampleProvider', function() {...});

// Service

sampleApp.provider('SampleService', function() {...});

// Controller

sampleApp.controller('SampleListCtrl', function() {...});

sampleApp.controller('SampleViewCtrl', function() {...});

sampleApp.controller('SampleCreateCtrl', function() {...});

**File Name**

* angular JS의 module 은 모듈명을 Under score(\_) 로 변환. SampleApp > sample\_app.js

/member/member\_app.js

/case\_study/case\_study\_app.js

**Template (HTML)**

* File 과 Folder 모두 동일하게 소문자 Under score(\_) 구분의 명사 조합.
* Folder 는 단위 프로그램 기준으로 최소단위의 단어 사용.
* IDE 의 Syntax Highlight 를 위해 확장자는 `.html` 사용.
* File은 File 명 만으로도 해당 프로그램을 알 수 있게 `프로그램명\_액션명.html` 조합으로 작성.

/member/member\_list.html

/member/member\_form.html

/case\_study/case\_study\_list.html

/case\_study/case\_study\_view.html

/case\_study/case\_study\_result.html

**변수 (공통)**

* 지역변수는 소문자로 시작하는 cameCase Style의 명사의 조합.
  + `count / memberAuth / caseStudy / isValid / checkYn`
* 전역변수는 대분지 Under score(\_) 구분의 명사의 조합.
  + `SITE\_CODE / SYSTEM\_ID / REAL\_FILE\_PATH`