



## 1. Programming Model Overview

The FW follows a POJO approach, where classes (DomainObjects, DomainServices, ViewModels), fields (Properties), and methods (Actions) are amended by means of annotations. Plus some 'supporting methods'.

### 1.1. Annotations

They are implemented in:

- o.a.i.applib.\* , some in
- javax.jdo.annotations. And of course
- javax.inject.Inject for dependency injection.

Applib annotations refer to enumerations, here listed in separately.

### 1.2. Supporting Methods

Some 'Supporting Methods' need to follow strict naming rules, i.e. need to have a certain prefix (disable, hide, validate) followed by a camelcased property or action name. title() is somewhat special ...

## 2. Remarks

- Enumeration values have been extracted from applib annotations and summarised into a table of its own (results in less redundancy and a more compact layout). References 'within' annotations list their respective default values.
- JDO annotations are grouped in two blocks - one contains those that refer to classes, the other refers to properties. Not complete yet.

## 3. Open issues

- @Property has CommandReification, CommandExecuteIn, CommandPersistence. What is its use?
- Will there be a kind of @Meta for 2.0.0 to wrap/combine JDO annotations?
- boolean flags in annotations will be removed and replaced by ...

- Are references to `@Pattern` outdated and can completely be replaced by `@Property.regexPattern*` ?

## 4. References

[1] Coad: Modeling in Color

| Coad Color | Coad Description                 | Color here | here used for |
|------------|----------------------------------|------------|---------------|
| pink       | moment-intervals                 | red        | method        |
| green      | entities (party/<br>place/thing) | green      | class         |
| yellow     | roles                            | orange     | property      |
| blue       | descriptions                     | blue       | layout        |

