# Common Lisp Object System (CLOS) Cheat Sheet

### ◆ CLOSの5つの特徴

*** クラスとメソッドは独立した概念**
*** クラスは多重継承ができる**
*** メソッドは多重ディスパッチができる**
*** メソッドは結合ができる**
*** 動的である**

### ◆ クラスの定義とオプション

```
(defclass class-name (classes…)
  ((slot1 :options opt-value)…))
```

| | |
|---|---|
| :initarg | 初期値の設定キーワード |
| :initform | デフォルト値 |
| :reader | 読み込み専用関数 |
| :writer | 書き込み専用関数 |
| :accessor | 読み書き両用関数 |
| :type | スロットのデータ型 |
| :allocation | :classまたは:instance |
| :documentation | ドキュメント文字列 |

### ◆ クラスに関するオペレータ

| | |
|---|---|
| `(with-slots (slot1 slot2) instance1 body…)` | インスタンスのスロットを変数に束縛するマクロ（ANSI標準） |
| `(let+ (((&slots slot1 slot2) instance1)) body…)` | let-plusパッケージによるスロットの束縛 |
| `(match instance1 ((class class-name slot1 slot2) body…))` | optimaパッケージによるスロットのマッチング |
| `(make-instance 'class-name :slot1-init v1 :slot2-init v2)` | インスタンスの生成 |
| `(change-class instance1 'class-name)` | クラスの変更 |
| `(class-of instance1)` | クラスの取得 |
| `(class-name class1)` | クラス名の取得 |

CLOSにおいて「クラス」とはデータ構造の一種です。複数のスロットをまとめて扱うことができ、多重継承に対応しており、複雑な構造体として機能することができます。

クラスを扱う場合、大きく3ステップ必要です。

1. defclassマクロでクラス（スロットの構造）を定義します
2. initialize-instance総称関数を再定義し、初期化処理を提供します
3. make-instance総称関数でクラスからインスタンスを生成します

※必要に応じて、print-object総称関数を再定義して、独自のクラスに対応する表示処理を提供します。

### ◆ メソッドの定義

```
(defmethod method-name ((var1 class-type) …)
  body…)
```

```
(defgeneric method-name (var1…)
  (:documentation "Documentation Strings…")
  (:method ((var1 class-type)…)
    body…))
```

同名のメソッドは総称関数(generic-function)に束ねられます。defgenericによる総称関数の定義を省略してdefmethodだけで定義した場合も、自動で総称関数が定義されます。

Common Lispは総称関数が呼び出された場合、登録されているメソッドから多重ディスパッチの方法により適切なメソッドを選び出し、必要であればそれらを結合して適用します。

### ◆ 標準メソッド結合

| | |
|---|---|
| standard（無指定時） | 上書き |
| + | 加算 |
| and, or | 論理積、論理和 |
| append, nconc | リスト結合 |
| list | リスト化 |
| max, min | 最大値、最小値 |
| progn | 単純実行 |

### ◆ 補助メソッド結合

| | |
|---|---|
| :around | 最初に呼ぶ |
| :before | 特定度順に呼ぶ |
| :after | 特定逆順に呼ぶ |

:around補助メソッドは最初に優先して特定度順に呼ばれます。次に、メソッドの特定度順に:beforeメソッド、基本メソッド、:afterメソッドが呼ばれますが、:afterメソッドだけは特定度の逆順で呼ばれる点に注意が必要です。

### ◆ 代表的な総称関数

| | |
|---|---|
| make-instance | インスタンス生成 |
| initialize-instance :after | コンストラクタ |
| print-object | 表示メソッド |

総称関数はメソッドの集合で、中身のメソッドは結合されます。メソッド結合の方式を指定しない場合はstandard（上書き戦略）が採用されるため、もっとも特定的なメソッド1つが適用されます。

CLOSにおいてメソッドはクラスから独立して定義されますが、メソッドの集合体である総称関数は引数の型に応じて多重ディスパッチの機能を提供するため、引数の型に合ったメソッドを自動で選択します。オブジェクトが自身の振る舞い（メソッド）を知っているというより、総称関数がオブジェクトに応じて動作を振り分ける、というイメージです。