**Object-Oriented Software Engineering**
Using UML, Patterns, and Java

# Chapter 8, Object Design: Object Constraint Language

## Outline of the Lecture

- OCL
- Simple predicates
- Preconditions
- Postconditions
- Contracts

## OCL Basic Concepts

- OCL expressions
  - Return **True** or **False**
  - Are evaluated in a specified context, either a class or an operation
  - All constraints apply to all instances.

## OCL Simple Predicates

Example:

```
context Tournament inv:
  self.getMaxNumPlayers() > 0
```

In English:

"The maximum number of players in any tournament should be a postive number."

Notes:

- "self" denotes all instances of "Tournament"
- OCL uses the same dot notation as Java.

## OCL Preconditions

Example:

```
context Tournament::acceptPlayer(p) pre:
     not self.isPlayerAccepted(p)
```

In English:

"The acceptPlayer(p) operation can only be invoked if player p has not yet been accepted in the tournament."

Notes:

- The context of a precondition is an operation
- isPlayerAccepted(p) is an operation defined by the class Tournament.

## OCL Postconditions

Example:

```
context Tournament::acceptPlayer(p) post:
  self.getNumPlayers() =
      self@pre.getNumPlayers() + 1
```

In English:

“The number of accepted player in a tournament increases by one after the completion of acceptPlayer()”

Notes:

- self@pre denotes the state of the tournament before the invocation of the operation.
- Self denotes the state of the tournament, in the *post* condition, after the completion of the operation.

# OCL Contract for acceptPlayer() in Tournament

```
context Tournament::acceptPlayer(p) pre:
  not isPlayerAccepted(p)

context Tournament::acceptPlayer(p) pre:
  getNumPlayers() < getMaxNumPlayers()

context Tournament::acceptPlayer(p) post:
  isPlayerAccepted(p)

context Tournament::acceptPlayer(p) post:
  getNumPlayers() = @pre.getNumPlayers() + 1
```

## OCL Contract for removePlayer() in Tournament

```
context Tournament::removePlayer(p) pre:
  isPlayerAccepted(p)

context Tournament::removePlayer(p) post:
  not isPlayerAccepted(p)

context Tournament::removePlayer(p) post:
  getNumPlayers() = @pre.getNumPlayers() - 1
```

## JavaDoc

- Add documentation comments to the source code.
- A doc comment consists of characters between **/**** and ***/**
- When JavaDoc parses a doc comment, leading * characters on each line are discarded. First, blanks and tabs preceding the initial * characters are also discarded.
- Doc comments may include HTML tags
- Example of a doc comment:

```
/**
* This is a <b> doc </b> comment
*/
```

## More on Java Doc

- Doc comments are only recognized when placed immediately **before class, interface, constructor, method or field declarations**.
- When you embed HTML tags within a doc comment, **you should not use heading tags such as <h1> and <h2>**, because JavaDoc creates an entire structured document and these structural tags interfere with the formatting of the generated document.

# Java Implementation of  Tournament class (Contract as a set of JavaDoc comments)

```
public class Tournament {
/** The maximum number of players
 * is positive at all times.
 * @invariant maxNumPlayers > 0
 */
private int maxNumPlayers;

/** The players List contains
 *  references to Players who are
 *  are registered with the
 *  Tournament. */
private List players;

/** Returns the current number of
 * players in the tournament. */
public int getNumPlayers() {…}

/** Returns the maximum number of
 * players in the tournament. */
public int getMaxNumPlayers() {…}
```

```
/** The acceptPlayer() operation
 * assumes that the specified
 * player has not been accepted
 * in the Tournament yet.
 * @pre !isPlayerAccepted(p)
 * @pre getNumPlayers()<maxNumPlayers
 * @post isPlayerAccepted(p)
 * @post getNumPlayers() =
 *     @pre.getNumPlayers() + 1
 */
public void acceptPlayer (Player p) {…}

/** The removePlayer() operation
 * assumes that the specified player
 * is currently in the Tournament.
 * @pre isPlayerAccepted(p)
 * @post !isPlayerAccepted(p)
 * @post getNumPlayers() =
 *     @pre.getNumPlayers() - 1
 */
public void removePlayer(Player p) {…}

}
```