Salesforce GROW

Module : Apex Language Constructs and Collections

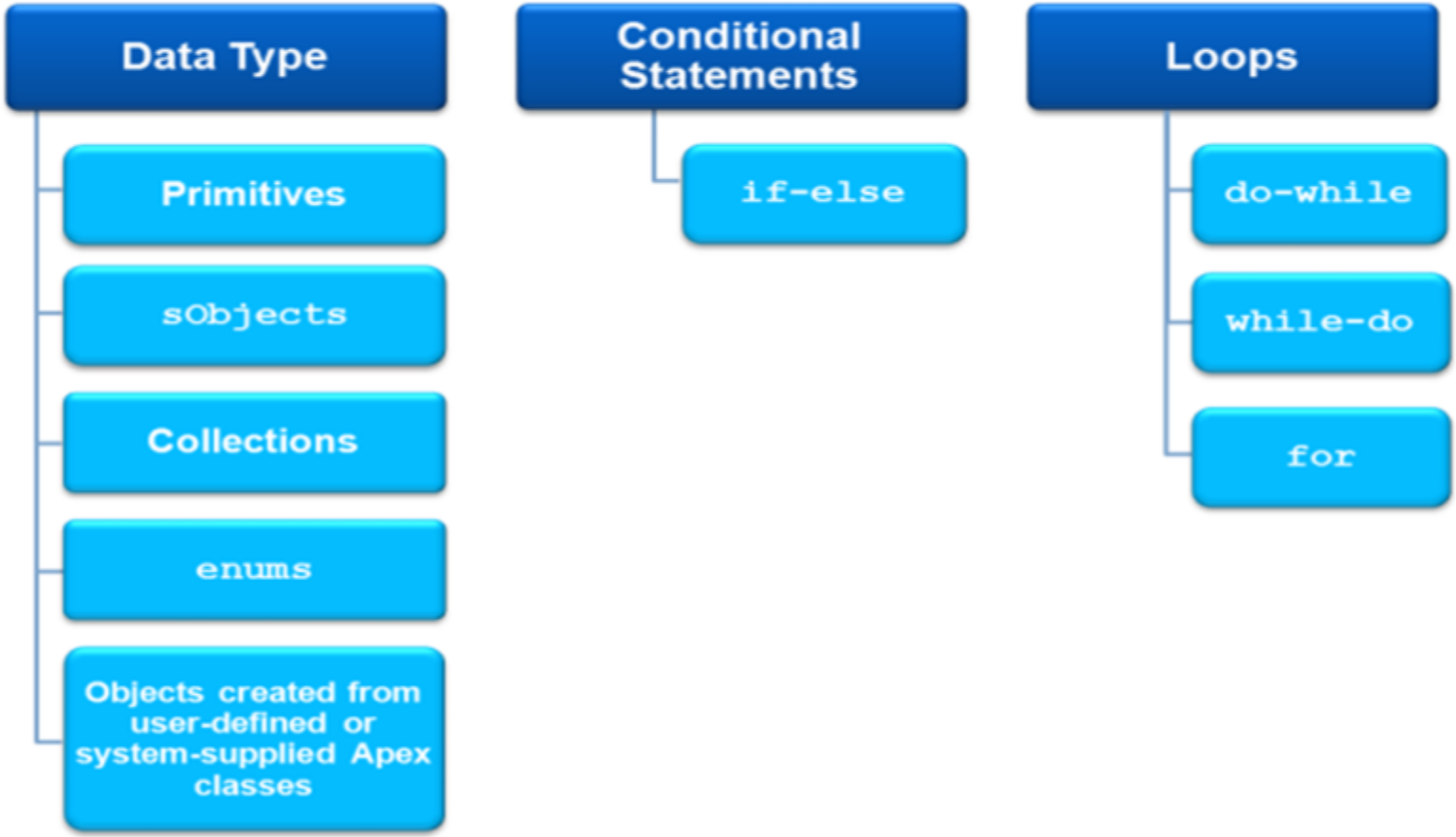
OurUniversity
EFMD CLIP ACCREDITED

Apex Language Constructs and Collections

Capgemini

Apex Language Constructs

Apex supports many basic language constructs that are similar to those in other programming languages.



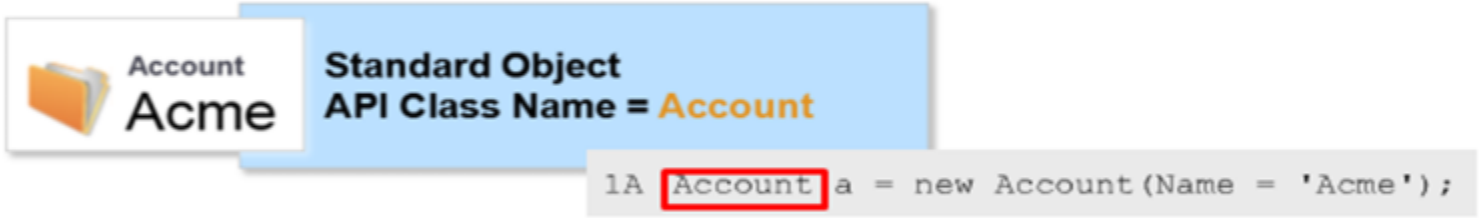
Primitive Data Types

Apex supports the following primitive data types:

- Integer
- Blob
- Boolean
- Date
- DateTime
- Decimal
- Double
- ID
- Long
- String
- Time

sObject Data Type

sObject: A generic data type that is the parent class for all standard and custom objects in Apex.



Comparing Collections

	Lists	Sets	Maps
Contain	Indexed values	Values	Key-value pairs
Ordered	Yes	No	No
Unique Values	No	Yes	No, but keys must be unique
Use case	Store the results of a SOQL query	Lookup criteria for relationship-based SOQL query	Cache of records indexed by ID
Notable methods	▪ `addAll()` ▪ `Sort()`	▪ `addAll()` ▪ `containsAll()` ▪ `retainAll()` ▪ `removeAll()`	▪ `containsKey()` ▪ `keySet()` ▪ `putAll()` ▪ `values()`

Enumerations Data Type

An enumeration is an abstract data type with values that each take on exactly one of a finite set of identifiers as specified.

Enumerations are typically used to define a set of possible values that don't otherwise have a numerical order, such as the suit of a card, or a particular season of the year.

Each value in the enumeration corresponds to an Integer value, starting from zero and incrementing by one from left to right.

Use "enum" keyword to specify set of constants.

Define Enumeration:

```
public enum Season {WINTER, SPRING, SUMMER, FALL}
```

Use Enumeration:

```
Season e = Season.WINTER;

Season m(Integer x, Season e) {

    if (e == Season.SUMMER) return e;
     //...
}
```

Objects created from user-defined or System-supplied apex classes

Data Types from Objects: There are two types of objects can be used in Apex as data type:

Standard Objects: These are objects included with Salesforce, by default, such as accounts, contacts, or opportunities.

Custom Objects: These are new objects user create to store information unique to your organization. Custom objects extend the functionality that standard objects provide.

Data Types from System-supplied Apex classes: Salesforce provided some System-supplied Apex classes as below, which can be used when building custom Visualforce controllers and controller extensions.

ApexPages: To add and check for messages associated with the current page, as well as to reference the current page.

Action Class: To create an action method that you can use in a Visualforce custom controller or controller extension

PageReference: A PageReference is a reference to an instantiation of a page.

SelectOption: A SelectOption object specifies one of the possible values for Visualforce selectCheckboxes, selectList or selectRadio component.

StandardController: Use a StandardController when defining an extension for a standard controller.

Message: Contains validation errors that occur when the end user saves the page when using a standard controller.

Conditional Statements

The conditional statement in Apex works similarly to Java:

```
if ([Boolean_condition])
    // Statement 1
else
    // Statement 2
```

is equivalent to:

```
Integer x, sign;
// Your code
if (x <= 0) {
    if (x == 0) {
            sign = 0;
    } else  {
            sign = -1;
    }
}
```

The else portion is always optional, and always groups with the closest if. For example:

```
Integer x, sign;
// Your code
if (x <= 0) if (x == 0) sign = 0; else sign = -1;
```

Repeated else if statements are also allowed. For example:

```
if (place == 1) {
    medal_color = 'gold';
} else if (place == 2) {
    medal_color = 'silver';
} else if (place == 3) {
    medal_color = 'bronze';
} else {
    medal_color = null;
}
```

Iterations(Loops)

Apex supports the following five types of procedural loops:

- do {*statement*} while (*Boolean_condition*);
- while (*Boolean_condition*) *statement*;
- for (*initialization*; *Boolean_exit_condition*; *increment*) *statement*;
- for (*variable* : *array_or_set*) *statement*;
- for (*variable* : [*inline_soql_query*]) *statement*;

All loops allow for loop control structures:

- break; exits the entire loop
- continue; skips to the next iteration of the loop

Hands-on Exercise

Apex: Data Types and Logic Exercise Guide
Exercise 2-1: Writing Basic Anonymous Blocks with Apex Data Types

https://lms.cfs-api.com/v1/content/8f644ab0-76e6-4f49-9d28-209b1b98afdd/presentation_content/external_files/DataTypesandLogicExerciseguide.pdf