調査 5 モデルカリキュラムの提言 コースウェア

15. Light Weight Language に関するスキル

Ⅰ. 概要	コードの作成や修正が容易とされる軽量プログラミング言語（Light Weight Language）を学習し、アプリケーション開発の手法を習得する。 本カリキュラムでは、まず Light Weight Language に位置づけされる Perl、PHP、Python、Ruby といったプログラミング言語の特徴について解説し、動作環境の構築や基本的な構文について習得する。 次に、前述した言語のうち、Ruby の焦点をあて、Ruby によるオブジェクト指向開発やフレームワークによる Web アプリケーション開発について学習する。最後に、オープンソースとして公開されている Web アプリケーションをカスタマイズし、機能を追加する方法について学習する。
Ⅱ. 対象専門分野	職種共通
Ⅲ. 受講対象者、 受講前提	プログラミング経験があることが望ましい。 なお、カリキュラム構成として一度に多くの言語を学ぶ構成であるため、ワークショップの開発実習に十分な配慮が必要である。
Ⅳ. 学習目標	• Light Weight Language の概要や基礎知識、各言語における特徴について理解する。 • Perl、PHP、Python、Ruby といった Light Weight Language の動作環境の設定方法、プログラムの基本構文について理解する。 • Ruby の組み込みクラスの利用方法について習得する。 • Ruby によるオブジェクト指向プログラミングの理解を目指す。 • Rails フレームワークを利用したアプリケーション開発（データベースアプリケーション、Web アプリケーション、プラグイン導入、テスト）について学習する。 • Light Weight Language で開発されたオープンソースシステムのパッケージを導入し、そのカスタマイズ方法について学習する。
Ⅴ. 使用教科書、 教材等	『プログラミング Ruby－達人プログラマーガイド』 David Thomas／Andrew Hunt 著、ピアソンエデュケーション刊 『Ruby on Rails 入門－優しい Rails の育て方』 西和則著、秀和システム刊

Ⅵ. 習得スキル の評価方法	講義終了後の受講レポート、定量アンケート、知識確認ミニテスト、 演習問題の取り組み状況を総合的に判断して評価を行う。
Ⅶ. カリキュラム の構成	レベル 1　第 1 回～第 10 回 レベル 2　第 11 回～第 15 回

講座内容

第 1 回 Light Weight Language の基本(講義＋ワークショップ 90 分)

Light Weight Language プログラミングの概要として、歴史や特徴、適用範囲について学習する。また、プログラムを開発する流れについて習得する。

(1) Light Weight Language の説明

1. Light Weight Language とは(概要、歴史、開発の経緯)
2. Light Weight Language の種類(Perl、PHP、Python、Ruby、等)
3. Light Weight Language と Heavyweight Language の特徴
   (利用用途による種別、各言語のメリット・デメリット)

(2) LightWeight Language による開発の流れ

1. 動作環境の構築(Perl、PHP、Python、Ruby)
2. エディタによるプログラム作成(Eclipse プラグインの利用)
   - Perl(EPIC : Eclipse Perl Integration)
   - PHP(PHPeclipse)
   - Python(PyDev : Python Development Environment)
   - Ruby(RDT : Ruby Development Tools)
3. プログラムと文字コード
4. 実行までの流れ
   - ローカルでの実行(コマンドラインでの実行)
   - Web アプリケーションでの実行
5. サンプルプログラムの作成

第 2 回　Perl の基本構造（講義＋ワークショップ　90 分）

Perl の基本的な仕組みと構成、基本構文を学習し、プログラミングの内容を理解する。
また、サンプルプログラムをもとに、簡単なプログラムの流れを確認する。

(1)　Perl の書き方の特徴

1. Perl とは（概要、言語開発の背景、バージョンによる差異、等）
2. Perl の特徴
3. Perl の動作方法（環境構築、ローカルアプリケーションとしての動作方法、Web アプリケーションとしての動作方法）
4. 拡張子の説明（pl、pm、cgi）
5. 開発エディタの使用方法（EPIC : Eclipse Perl Integration）
6. Perl の基本構造と文法（プログラムの構成と言語仕様）
7. Perl のライセンス形態
8. Perl による開発事例
9. 公式サイトとリファレンス紹介

(2)　基本的なプログラム記述の例

1. 変数、定数、配列
2. 演算子（代入、算術、比較、論理、文字列比較、文字列置換、等）
3. 制御構造（条件分岐処理、繰り返し処理）
4. 標準入出力
5. 正規表現
6. サブルーチンとライブラリ
7. 組み込みモジュール
8. サンプルプログラム（標準出力による実行例）

第 3 回　PHP の基本構造（講義＋ワークショップ　90 分）

PHP の基本的な仕組みと構成、基本構文を学習し、プログラミングの内容を理解する。

また、サンプルプログラムをもとに、簡単なプログラムの流れを確認する。

(1)　PHP の書き方の特徴

1. PHP とは（概要、言語開発の背景、バージョンによる差異、等）
2. PHP の特徴
3. PHP の動作方法（環境構築、ローカルアプリケーションとしての動作方法、Web アプリケーションとしての動作方法）
4. 開発エディタの使用方法（PHPeclipse）
5. PHP の基本構造と文法（プログラムの構成と言語仕様）
6. PHP のライセンス形態
7. PHP による開発事例
8. 公式サイトとリファレンス紹介

(2)　基本的なプログラム記述の例

1. 変数、定数、配列
2. 演算子（代入、算術、比較、論理、等）
3. 制御構造（条件分岐処理、繰り返し処理）
4. 例外処理（try ブロック、catch ブロック、throw 文）
5. ユーザ定義関数と標準関数
6. 拡張モジュール（php.ini での設定、等）
7. サンプルプログラム（リダイレクト処理による実行例）

第 4 回 Python の基本構造（講義＋ワークショップ 90 分）

Python の基本的な仕組みと構成、基本構文を学習し、プログラミングの内容を理解する。
また、サンプルプログラムをもとに、簡単なプログラムの流れを確認する。

(1) Python の書き方の特徴

1. Python とは（概要、言語開発の背景、バージョンによる差異、等）
2. Python の特徴
3. Python の動作方法（環境構築、ローカルアプリケーションとしての動作方法、Web アプリケーションとしての動作方法）
4. 開発エディタの使用方法（PyDev : Python Development Environment）
5. Python の基本構造と文法（プログラムの構成と言語仕様）
6. Python のライセンス形態
7. Python による開発事例
8. 公式サイトとリファレンス紹介

(2) 基本的なプログラム記述の例

1. 変数、定数、配列
2. 演算子（代入、算術、比較、論理、等）
3. 制御構造（条件分岐処理、繰り返し処理）
4. 例外処理（try-except）
5. ユーザ定義関数と組み込み関数
6. 外部ライブラリを利用した GUI の作成（Tkinter）
7. サンプルプログラム（CGI による実行例）

第 5 回 Ruby の基本構造(講義＋ワークショップ 90 分)

Ruby の基本的な仕組みと構成、基本構文を学習し、プログラミングの内容を理解する。
また、サンプルプログラムをもとに、簡単なプログラムの流れを確認する。

(1) Ruby の書き方の特徴

1. Ruby とは(概要、言語開発の背景、バージョンによる差異、等)
2. Ruby の特徴
3. Rubyの動作方法(環境構築、ローカルアプリケーションとしての動作方法、Web アプリケーションとしての動作方法)
4. 開発エディタの使用方法(RDT : Ruby Development Tools)
5. Ruby の基本構造と文法(プログラムの構成と言語仕様)
6. Ruby のライセンス形態
7. Ruby による開発事例
8. 公式サイトとリファレンス紹介

(2) 基本的なプログラム記述の例

1. 変数、定数
2. 演算子(代入、算術、比較、論理、範囲、等)
3. 制御構造(条件分岐処理、繰り返し処理)
4. 例外処理(rescue、Exception)
5. メソッド(モジュール関数、定義、戻り値の設定、呼び出し)
6. 組み込みライブラリ(組み込み関数、組み込み変数、組み込みクラス、等)
7. ブロック構文
8. サンプルプログラム(標準出力)

第 6 回　オブジェクト指向プログラミング（講義＋ワークショップ　90 分）

Ruby のクラスの使用方法を学習することで、オブジェクト指向の考え方を習得する。

また、サンプルプログラムを作成することで、オブジェクト指向プログラミングの利点を学習する。

(1)　オブジェクト指向の説明

1. オブジェクト指向とは
2. オブジェクト指向の特徴（継承、カプセル化、ポリモフィズム）と利点

(2)　オブジェクト指向プログラミングの説明

1. オブジェクト指向プログラミングとは
2. クラスの定義（クラス定義、特異クラス定義、モジュール定義）
3. メソッドの定義（メソッド定義、特異メソッド定義、クラスメソッド定義）
4. クラスの継承（継承関係、ダックタイピング）
5. メソッドの呼び出し制限（カプセル化：情報隠蔽）
6. ポリモフィズム（オーバーライド、キャスト）
7. Mix-in（クラスにモジュールの取り込み、include メソッド）
8. サンプルプログラム（クラス使用方法）

第 7 回　組み込みクラス[データ構造](講義＋ワークショップ　90 分)

Ruby に組み込まれている、配列やハッシュといったデータ構造に特化したクラスについて学習する。また、サンプルプログラムを通して、データ構造に関するクラスの使用方法を習得する。

(1)　データ構造の説明
   1. データ構造とは(概要、種類)
   2. データ構造に関する組み込みクラス(Array、Hash、Struct、等)

(2)　配列の説明
   1. 配列とは(概要、変数との違い、Array クラスの説明)
   2. Array クラスの構成(メソッドの種類、等)
   3. Array クラスの操作方法(要素取得、結合、ソート)
   4. サンプルプログラム

(3)　ハッシュの説明
   1. ハッシュとは(概要、Hash クラスの説明)
   2. Hash クラスの構成(メソッドの種類、等)
   3. Hash クラスの操作方法(ハッシュの生成、キーの設定、ハッシュサイズの取得)
   4. サンプルプログラム

(4)　構造化クラスの説明
   1. 構造化クラスとは(概要、構造化クラスの説明)
   2. Struct クラスの構成(メソッドの種類、等)
   3. Struct クラスの操作方法
   4. サンプルプログラム

第 8 回　組み込みクラス[データ操作](講義＋ワークショップ　90 分)

Ruby に組み込まれている数値や文字列、日付といったデータ操作に特化したクラスについて学習する。また、サンプルプログラムを通じて、データ操作に関するクラスの使用方法を習得する。

(1)　データ操作の説明
1. データ操作とは(概要、種類)
2. データ操作に関する組み込みクラス(Numeric、String、Time、等)

(2)　数値の説明
1. 数値とは(概要、数値の扱い、Numeric クラスの説明)
2. Numeric クラスの構成(メソッドの種類、等)
3. Numeric クラスの操作方法(算術演算、数値型変換、ビット演算)
4. サンプルプログラム

(3)　文字列の説明
1. 文字列とは(概要、文字列の扱い、String クラスの説明)
2. String クラスの構成(メソッドの種類、等)
3. String クラスの操作方法(文字分割、文字比較、文字検索／置換)
4. サンプルプログラム

(4)　その他データ操作のクラスの紹介
1. 日付／時刻の操作(Time クラス)
2. シンボルを表すクラス(Symbol クラス)
3. 正規表現による文字のパターンマッチング(Regexp クラス)

第 9 回　組み込みクラス[ファイル管理]（講義＋ワークショップ　90 分）

Ruby に組み込まれている、ファイルに対する入出力やファイル操作といった、ファイル管理に特化したクラスについて学習する。また、サンプルプログラムを通じて、ファイル管理に関するクラスの使用方法を習得する。

(1)　ファイル管理の説明
   1. ファイル管理とは（概要、種類）
   2. ファイル管理に関する組み込みクラス（File::Stat、IO、File、Dir、等）

(2)　ファイル情報の説明
   1. ファイル情報とは（概要、File::Stat クラスの説明）
   2. Fiile クラスに関する定数（File::Constants）
   3. File::Stat クラスの構成（メソッドの種類、等）
   4. File::Stat クラスの操作方法（ファイル情報の取得、等）
   5. サンプルプログラム

(3)　ファイル入出力の説明
   1. ファイル入出力とは（概要、IO クラスの説明）
   2. IO クラスの構成（メソッドの種類、等）
   3. IO クラスの操作方法（バイナリ入出力、テキスト入出力）
   4. サンプルプログラム

(4)　ファイル／ディレクトリ操作の説明
   1. ファイル／ディレクトリ操作とは（概要、File／Dir クラスの説明）
   2. File／Dir クラスの構成（メソッドの種類、等）
   3. File／Dir クラスの操作方法（ファイルコピー、ディレクトリ作成、等）
   4. サンプルプログラム

第 10 回 GUI アプリケーション開発(講義 90 分)

GUI アプリケーションを開発するために、GUI ライブラリである Ruby/Tk と Ruby-GNOME2 を使用する方法について学習する。

(1) Ruby/Tk による GUI アプリケーション開発
   1. Ruby/Tk とは(概要、導入方法)
   2. ウィジェットとイベント処理
   3. キャンバスによる描画処理
   4. Ruby/Tk によるサンプルプログラム

(2) Ruby-GNOME2 による GUI アプリケーション開発
   1. Ruby-GNOME2 とは(概要、導入方法)
   2. シグナルとコールバックの概念
   3. ウィジェットの種類とパラメータ
   4. GnomeCanvas2 による描画処理
   5. Ruby-GNOME2 によるサンプルプログラム

(3) GUI ライブラリの特徴と比較
   1. Ruby/Tk の特徴
   2. Ruby-GNOME2 の特徴
   3. その他 GUI ライブラリの特徴
   4. GUI ライブラリの比較

第 11 回 Ruby on Rails（講義 90 分）

Ruby を使用した Web アプリケーションを開発するフレームワーク Ruby on Rails の概念と構成といった仕組みについて学習する。また、Ruby on Rails で採用されているソフトウェア設計アーキテクチャである MVC アーキテクチャの仕組みについて習得する。

(1) Ruby on Rails の仕組み
   1. フレームワークとは
   2. Ruby on Rails とは（Rails の概要、ディレクトリ構成、設定ファイル、等）
   3. Rails フレームワークの構成（ActionMailer、ActionPack、ActiveRecord、ActiveSupport、ActionWebService、Railties）
   4. Ruby on Rails の導入方法（RubyGems）
   5. Rails の起動（WEBrick の概要、起動コマンド）

(2) コードジェネレートの説明
   1. コードジェネレートとは（概要）
   2. コードの自動生成（script/generate コマンド）
   3. scaffold のオプション（Model、View、Controller の生成）

(3) MVC アーキテクチャ
   1. ソフトウェア設計アーキテクチャの概要
   2. MVC アーキテクチャの概要
   3. Web アプリケーションフレームワークとしての Rails の構成要素（Model、View、Controller）

第 12 回　データベースアプリケーション開発（講義＋ワークショップ　90 分）

Rails フレームワークを使用したデータベースアプリケーション開発の流れについて学習する。Rails フレームワークの ActiveRecord を使用して、データベースへの操作を実現する方法を学習する。

(1)　データベースの仕組み

1. データベースとは（概要、種類、導入方法）
2. データベースの構成（テーブル、フィールド、データ型）
3. データベースの構築（テーブルの定義方法、レコードの追加方法）

(2)　ActiveRecord の説明

1. Rails によるデータベース連携とは
2. ActiveRecord とは（概要、構成）
3. データベースの連携（YAML の概要、記述方法、database.yml の修正）
4. Model の作成（Model のジェネレート、Model へのアクセス）

(3)　データベースアプリケーション開発

1. テーブルの定義
2. テーブル操作（追加、更新、削除、検索）の流れ
3. トランザクション処理の流れ
4. サンプルプログラム

第 13 回　Web アプリケーション開発（講義＋ワークショップ　90 分）

Rails フレームワークを使用した Web アプリケーション開発の流れについて学習する。また、Web アプリケーションフレームワークとしての Rails の構成要素（Model、View、Controller）の機能について学習する。

(1)　Rails による Web アプリケーション開発の説明

1. Web アプリケーション開発の流れとは
2. Rails の構成要素（Model、View、Controller）のジェネレート
3. データベースの設定（database.yml の設定）
4. Web サーバの起動と確認（WEBrick、ポート指定）

(2)　Web アプリケーションのカスタマイズ

1. ルーティングの定義（複数ページの遷移設定、等）
2. ユーザ表示箇所のカスタマイズ（View の書き換え、RHTML の概要、編集、CSS の利用）
3. 入力値の検証（概要、validates の定義）
4. リダイレクト処理（概要、メソッドの追加（Controller／Model））
5. サンプルプログラム

第 14 回　プラグイン導入と開発（講義＋ワークショップ　90 分）

Rails の機能を拡張する方法として、プラグインの導入について学習する。プラグインは、既存のものを使用する方法と、新規にプラグインを作成する方法について学習する。

(1)　プラグインとは

1. プラグインの概要（メリット）
2. プラグインの配置と構成
3. 既存プラグインの種類（例外情報保存プラグイン、RJS テストプラグイン、等）
4. プラグインのインストール（プラグインコマンド（script/plugin、等）の説明）
5. サンプルプログラム（プラグインの導入例）

(2)　新規プラグインの開発

1. 新規プラグインの作成の流れ（ジェネレータの実行、機能の実装、実行）
2. テスト環境の構築（テスト定義ファイルの作成、テストの実行）
3. サンプルプログラム（新規プラグインの導入例）

第 15 回 オープンソースシステムのカスタマイズ（講義＋ワークショップ 90 分）

オープンソースシステムとして公開されている Rubricks という CMS（コンテンツマネジメントシステム）を利用して、独自に作成したコンポーネントを導入することによってカスタマイズする方法について学習する。

(1) Rubricks とは
   1. CMS（コンテンツマネジメントシステム）の概要
   2. Rubricks の構成（Rubricks の機能、コンポーネントアーキテクチャ、ブロックアーキテクチャ、等）
   3. Rich Internet Applications（Ajax、等）
   4. コンポーネントの種類と導入方法（サードパーティコンポーネント、等）

(2) 新規コンポーネントの開発
   1. 新規コンポーネントの作成の流れ（ジェネレータの実行、機能の実装、実行）
   2. テスト環境の構築（テスト定義ファイルの作成、テストの実行）
   3. サンプルプログラム（新規コンポーネントの導入例）

(3) オープンソースシステムのカスタマイズ例
   1. オープンソースシステムの紹介（Xoops、suguarCRM、OpenPNE、等）
   2. オープンソースシステムのカスタマイズ方法（例）

以上