# **プログラムユニット設計書**

## **プログラムユニット詳細処理**

|  |  |
| --- | --- |
| **プログラムユニットID** | **searchFragment** |
| **プログラムユニット名** | フィールド |
| **引数** | なし |
| **戻り値** | なし |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  1．String型でジャンル、問題数、~~誰が作ったか~~、キーワードの値を入れる変数3つを宣言する。  String spener1(ジャンル)  String spener2(問題数)  ~~String spener3(誰が作ったか)~~  String keyword(キーワード) | |
| **制約事項** | |
| 【レイアウト】  //XMLファイルにスピナーを追加する×2  ジャンル =>スピナー（なし,。。。。）  問題数 =>スピナー（なし,01~10,11~20,21~30,31~40,41~50,51~60,61~70,71~80,81~90,91~100）  誰が作ったか（スピナー）  //xmlファイルにキーワード検索用のEditTextを追加  キーワード =>テキスト  xmlファイルに検索決定ボタンの追加 | |
| **備考** | |
| <https://qiita.com/QiitaD/items/e7b7ed95d0abaa106337> | |

## **プログラムユニット詳細処理**

|  |  |
| --- | --- |
| **プログラムユニットID** | **searchFragment** |
| **プログラムユニット名** | onClickEnter() |
| **引数** | なし |
| **戻り値** | なし |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・検索決定ボタンを押したときに起こる処理  1．検索欄に条件が入っているかの判定を行う  //条件が入っているかの判定  //検索欄（スピナー、テキスト）に何か1つが入ってたら  listUpPack()を起動する  //全部入ってなかったら（ジャンル、問題数が全部「なし」かつキーワードが未記入）  「検索欄に入力してください」というポップアップが表示される  ×を押すとポップアップが消える  　If(spener1 = “なし” && spener2 = “なし” && keyword = null){  「検索欄に入力してください」というポップアップが表示される  ×を押すとポップアップが消える  }else{  listUpPackを起動  } | |
| **制約事項** | |
|  | |
| **備考** | |
| ポップアップ  <https://blog.codecamp.jp/android-app-development-2> | |

## **プログラムユニット詳細処理**

|  |  |
| --- | --- |
| **プログラムユニットID** | **searchFragment** |
| **プログラムユニット名** | listUpPack() |
| **引数** |  |
| **戻り値** | なし |
| **処理内容** | |
| 1.int型でクイズ数上限とクイズ数下限の値を入れる変数2つを宣言。  int maxQuiz //クイズ数上限  int minQuiz //クイズ数下限  2.int型で”~”が何文字目に来るかの値を入れる変数1つを宣言。  int index //クイズ数”1~10”の中の”~”が何文字目か  3.検索情報に入れた条件に合ったパックを探す処理を1行ずつし、それを行数分繰り返す。  ファイルの行分繰り返し処理(for)  For文始まり～  　3.1対象のリストから”,”で区切られた情報のデータを変数として宣言。  **String**[] selectData = allLines.get(i).split(","); //“,”区切りのデータを入れる配列  //以下, 条件文がfalseの場合はその検索条件を満たす  3.2ジャンルが検索条件に合致するデータを探す  3.2.1ジャンルの部分が「なし」じゃないかつ、プレイヤーが選択したジャンルの文字列とリスト中のジャンルの文字列を比較し、違っていたら中断し次の行の検索へ  If(ジャンルが“なし”じゃない && spener1（ジャンル）<> selectData[0]) continue;  //ジャンルの入っているブロックがselectData[0]  // spener1（ジャンル）が検索条件と違ったらcontinue  3.3問題数が検索条件に合致するデータを探す  　3.3.1問題数の部分が「なし」じゃないかつ、プレイヤーが選択した問題数とリスト中の問題数を比較し、違っていたら中断し次の行の検索へ  index = str.indexOf("~")  maxQuiz=(int)クイズ数.substring(index+1)  //クイズ数.substring(index+1) = ~の後の数字  minQuiz=(int)クイズ数.substring(0, index+1)  //クイズ数.substring(index+1) = ~の前の数字  If(問題数が“なし”じゃない && (maxQuiz < selectData[1] || minQuiz > selectData[1])) continue;  //問題数の入っているブロックがselectData[1]  //問題数の範囲（maxQuiz~minQuiz）に検索条件（selectData[1]）が入っていなかったらcontinue  ~~If(自作かどうか指定されている && spener3（自作かどうか）<>selectDate[2]) continue;~~  ~~//自作かデフォルトかが入っているブロックがselectData[2]~~  ~~// spener3（自作かどうか）が検索条件と違ったらcontinue~~  3.4キーワードが検索条件に合致するデータを探す  3.4.1キーワード入力欄が空白じゃないかつ、プレイヤーが入力したキーワードの文字列がリスト中のタイトルと説明文の文字列に含まれていなかったら中断し次の行の検索へ  If(キーワード入力欄が空白じゃない && (!selectData[2].contains(keyword（キーワード）))) &&(!selectData[3].contains(keyword（キーワード）)) continue;  //タイトルが入っているブロックがselectData[2]  //説明文が入っているブロックがselectData[3]  //入力されたkeyword（キーワード）がタイトルと説明文に入っていなかったらcontinue  3.5すべての検索条件を満たしここまで来たパックの行数(=i)をselectLinesに加える  3.6すべての行を検索するまで続け、終わったら閉じる  continue  End for  3.7スタックしていたselectPackFragmentに戻り、searchFragmentを閉じる | |
| **制約事項** | |
|  | |
| **備考** | |
| 【Main Application】  //パック情報の入ったファイルの読み込み  //ファイルを読み込んでリストを生成し、その変数を宣言（検索機能後のリストとしてString selectLinesを宣言。  List<**String**> selectLines = Files.readAllList(path, Charset.forName("ファイル名")); | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **selectPackFragment** |
| **プログラムユニット名** | フィールド |
| **引数** | なし |
| **戻り値** | なし |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  int形で現在のページ、全ページ数の値を入れる変数2つを宣言する。  int pageCurrent  int pageAll | |
| **制約事項** | |
|  | |
| **備考** | |
|  | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **selectPackFragment** |
| **プログラムユニット名** | onCreate() |
| **引数** |  |
| **戻り値** |  |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・allListをselectListに代入することで次のshowPackListがonCreateの時も起きるようにした処理  1．selectListを初期化  2．selectListにallListを代入する  3・selectListを新着順に並び替え  4.pageAllにselectListの長さから導かれる全ページ数の値を代入する  pageAll（全ページ数）=(int)((selectListの長さ-1)/10)+1  5．pageCurrent（現在ページ）＝1を入れて初期化 | |
| **制約事項** | |
| 【レイアウト】  //xmlファイルに次のページに行くボタンを設置  //xmlファイルに戻るページに行くボタンを設置  //xmlファイルに検索機能ボタンを設置  //xmlファイルにページ数を記録するTextを設置  //xmlファイルにパックボタンを設置×10  スクロールバーを設置 | |
| **備考** | |
| 【Main Application】  //パック情報の入ったファイルの読み込み  //ファイルを読み込んでリストを生成し、その変数を宣言（すべてのパックが入ったリストとしてString allLinesを宣言。  List<**String**> allLines = Files.readAllLines(path, Charset.forName("ファイル名")); | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **selectPackFragment** |
| **プログラムユニット名** | onCreateView() |
| **引数** |  |
| **戻り値** |  |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・普通に来た時にはonCreateの次に起きる処理。searchFragmentから戻ってきた時にはこのメソッドから始まる  1．pageCurrent（現在ページ）＝1を入れて初期化  pageAll（全ページ数）=(int)((selectListの長さ-1)/10)+1  2．showPackListを起動。 | |
| **制約事項** | |
|  | |
| **備考** | |
|  | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **selectPackFragment** |
| **プログラムユニット名** | showPackList() |
| **引数** | なし |
| **戻り値** |  |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・デフォルト、もしくは検索結果によるパックのデータを表示する処理  （MainApplicationクラスにあるselectListに表示されるIDがある）  1．String型でタイトル、問題数、説明文の値を入れる変数をそれぞれ10個ずつ、合計30個宣言する  2．変数をレイアウト上に作成したボタン10個に3個ずつ設定する  3.selectListの（10×pageCurrent-9）番目～（10×pageCurrent）番目のパック情報をボタンに表示  3.1selectListから”,”で区切られた情報のデータを変数として宣言  　for文で（i = 10×pageCurrent-9; i <10×pageCurrent; i++）{  String[] data = selectList.get(i).split(",");　//“,”区切りのデータを入れる配列  その行のタイトル、問題数、説明文のデータを出力し、  それぞれ  createPack i- {10×(PageCurrent-1)} －1,  createPack i+1- {10×(PageCurrent-1)} －2,  createPack i+2- {10×(PageCurrent-1)} －3  に代入する  }  それぞれ変数をcreatePack1-1,1-2,1-3,（タイトル、問題数、説明文）、  createPack2-1……..とし、〇-1の欄にはタイトル、〇-2の欄には問題数、〇-3の欄には説明文が入るようにする。  ボタン内にタイトル、問題数、説明文の順で改行して設置する。  現在ページ数をTextに表示  ワンピース大全  30問  ワンピースのクイズだよ  タイトル  問題数  説明文  createPack3-1  createPack3-2  createPack3-3 | |
| **制約事項** | |
|  | |
| **備考** | |
|  | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **selectPackFragment** |
| **プログラムユニット名** | onClickSetPackId() |
| **引数** |  |
| **戻り値** | true,false |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・パックのボタンを押したときに起こる処理  makePackActivityからきたのか、takeQuizPackActivityからきたのか判断して、makePackActivityだったら編集画面へ、takeQuizPackActivityだったらクイズ解答へ。  押されたボタンに対応するパックIDをApplicationクラスに渡す  ApplicationクラスのfromMakePackActivityがtrueなら  editPackFragmentを呼ぶ  ApplicationクラスのfromMakePackActivityが falseなら  takeQuizFragmentを呼ぶ  フラグメントを閉じる | |
| **制約事項** | |
|  | |
| **備考** | |
| 【mainApplication】  Boolean型でmakePackActivityからきたかどうかを入れる変数を宣言。  　boolean fromMakePackActivity | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **selectPackFragment** |
| **プログラムユニット名** | onClickSearch() |
| **引数** | なし |
| **戻り値** | なし |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・検索機能ボタンを押したときに起こる処理  pageCurrent（現在ページ数）を1にする  searchFragmentを生成  selectPackFragmntをスタック | |
| **制約事項** | |
|  | |
| **備考** | |
|  | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **selectPackFragment** |
| **プログラムユニット名** | onClickNextPage() |
| **引数** | なし |
| **戻り値** | なし |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・次のページに行くボタンを押したときに起こる処理  pageAll（全ページ数）とpageCurrent（現在ページ数）が異なる場合  pageCurrent（現在ページ数）に1足す  showPackList()を呼び出す  pageAll（全ページ数）とpageCurrent（現在ページ数）が等しい場合  何もしない | |
| **制約事項** | |
|  | |
| **備考** | |
|  | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **selectPackFragment** |
| **プログラムユニット名** | onClickFrontPage() |
| **引数** | なし |
| **戻り値** | なし |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・前のページに行くボタンを押したときに起こる処理  pageCurrent（現在ページ数）が1ではない場合  pageCurrent（現在ページ数）から1を引く  showPackList()を呼び出す  現在ページ数が1の場合  何もしない | |
| **制約事項** | |
|  | |
| **備考** | |
|  | |

|  |  |
| --- | --- |
| **プログラムユニットID** | **makeNewPackFragment** |
| **プログラムユニット名** | onClickEnter() |
| **引数** | なし |
| **戻り値** | なし |
| **処理内容** | |
| *＊実装可能なレベルまで詳細化する*  ・パック情報決定ボタンを押すと起きる処理  スピナーの値(ジャンル)をmakePackAcitivityのジャンルを入れる変数に代入  タイトル名のテキストボックスの値をmakePackAcitivityのタイトル名を入れる変数に代入  説明文のテキストボックスの値をmakePackActivityの説明文を入れる変数に代入  //何か1つでも未記入の場合、  If(packTitle=null || packIntroduction =null　){  「すべての欄に入力してください」というポップアップを表示  ×を押してポップアップが消える  }else{  そうじゃなかったら、makeNewPackFragmentをスタックしてeditQuizFragmentを起動させる。  } | |
| **制約事項** | |
| 【レイアウト】  //xmlファイルにスピナーを追加する×1  //xmlファイルにEditTextを追加×2  //xmlファイルにパック情報決定ボタンの追加 | |
| **備考** | |
| 【makePackActivity】  String型でジャンル、タイトル、説明文の値を入れる変数3つを宣言する。  String packGenre  String packTitle  String packIntroduction | |