知能システム演習Ｂ
りんごキャッチゲーム その２

2014.7.1

小方研究室

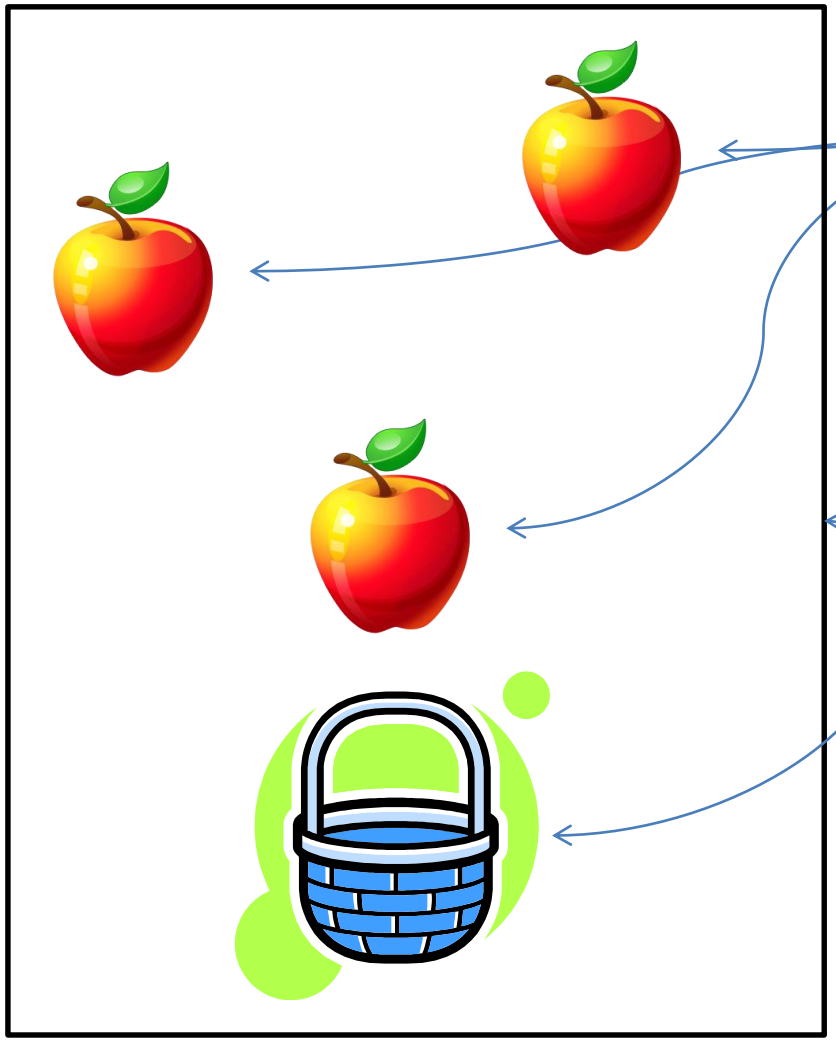
目次

- 前回の概要
- りんごキャッチのインタフェース

前回の概要(1/3)

- 籠を操作することで落下してくるリンゴをキャッチ
- ゲームの要素
  - リンゴ
    - 一つ以上
    - 落下速度一定，落下タイミング不定
  - 籠
    - 一つ
    - 操作可能

前回の概要(2/3)



「リンゴ」クラス
　値：座標、落下開始時間、滞空時間
メソッド：落下

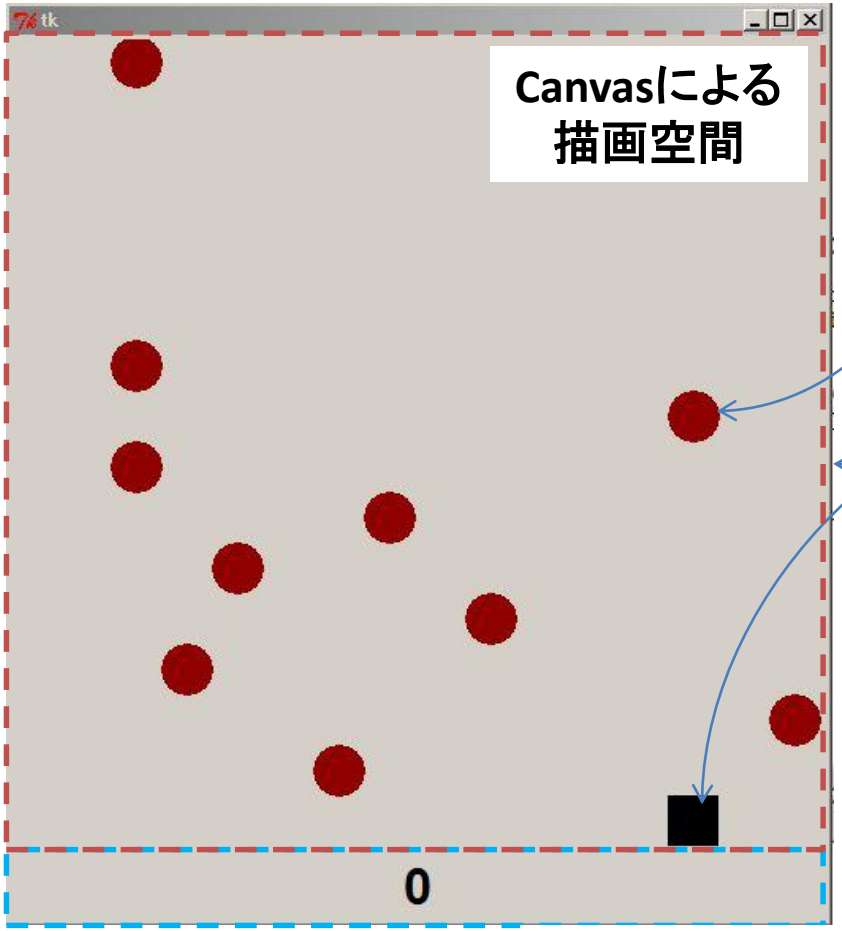
「かご」クラス
　値：座標，りんご獲得数
メソッド：移動，（待機）

「空間」クラス
　値：経過時間，空間全体
メソッド：状態更新，画面更新
　　入力待ち（メインループ）

前回の概要(3/3)

- 入力があるまで待機，入力された場合以下の処理
  - 時間変数を一つぶん増加
  - 籠の座標の移動または待機処理
  - リンゴの落下処理
  - キャッチ判定
  - 籠やリンゴの状態に基づき画面更新

りんごキャッチのインタフェース

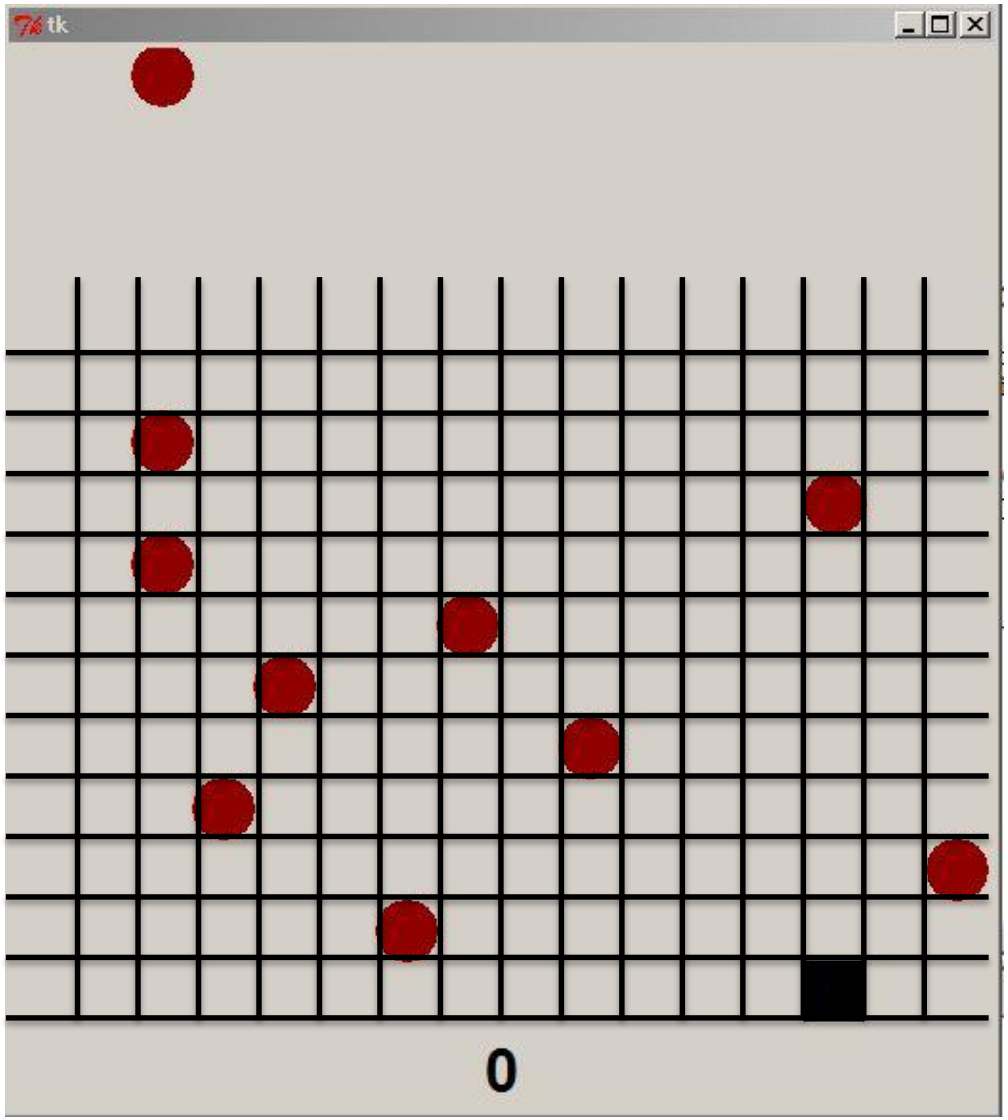


「リンゴ画像」
リンゴの数だけ，りんご画像(楕円描画で代用)のインスタンスを用意

「かご画像」
かご画像(四角形描画で代用)インスタンスを用意

「描画ウィンドウ」
親ウィンドウを用意し，Canvas()とLabel()により描画空間を確保

内部データと実際の描画



リンゴの内部データの座標管理
x=0,1,2,3,4,5…/y=0,1,2,3,4,5…

ゆっくりと落下していく

マス目に沿った落下

1マス＝32×32(px)と仮定
座標データに32を掛け算
ウィンドウの高さや幅も
「32×マスの数」で計算

インターフェースのまとめ

#*必要な数値準備**
　マスの大きさ，縦のマスの数，横のマスの数，リンゴの数
#*内部データの準備**
　apple_falling = worldクラス
#*インターフェースの準備**
#操作範囲の描画**
　親ウィンドウの用意
　親ウィンドウに対して，描画枠の準備（Canvas）
#*籠
cage =かご画像インスタンスの用意
#*リンゴ
#リンゴ画像インスタンス保存用のリスト
　apple_img = []
　for i in リンゴの数だけ:
　　i番目のリンゴ画像インスタンスの用意，
#リンゴ獲得数**
　親ウィンドウに対して，獲得数表示枠の準備（Label）
#*キーボード操作と移動処理メソッドのバインディング**
　root.bind(‘<Key>’, move);;;*別に「def move(event)」という操作に対する処理の関数を定義*