Webプログラミング実習1

2017/7/26

Kazuma Sekiguchi
class@cieds.jp

Canvasでアニメーション

- 実のところ、結構大変なので、あまりオススメしない
  - CSSアニメーションの方が速度が出ることもあり、それほど利用されている場面も少ない
- Canvas自体にはアニメーション機能は備わっていない
  - JSでCanvasにオブジェクトを描画する
  - JSでCanvasに描画したオブジェクトを消去する
  - JSでCanvasにオブジェクトを少し動かして描画する

  の繰り返し
  - setIntervalやrequestAnimationFrameを多用する

setInterval,setTimeout, requestAnimationFrame

- setIntervalは指定した時間が経過するごとに関数を呼び出す機能
  - setTimeoutは指定した時間後に関数を呼び出す機能
  - setTimeoutで呼び出した関数内で更にsetTimeoutを呼び出せば、永久ループが可能
- requestAnimationFrameはアニメーションが可能になったら実行される関数
  - どういう間隔で実行されるかはブラウザーや環境などに依存する
  - （ある程度環境が整っていれば）スムーズな動きが実現可能
  - Canvasアニメーションでは最近はこれを利用する率が高い（ブラウザーがすべて対応していない点注意）

アニメーション

- 描画自体は比較的簡単
  - 上手くランダム関数などを利用すると毎回違った画像になるので、変化を与えることが可能
- アニメーション
  - 一度描画したものを消去して新しく描画し直す
    - これを高速に繰り返せば、アニメーションしているようになる
    - 実際のところ、これはものすごく大変
    - 同じ図形を動かすだけならそれほどでもないが、ループで複数作成した場合は難しい

jCanvasとアニメーション

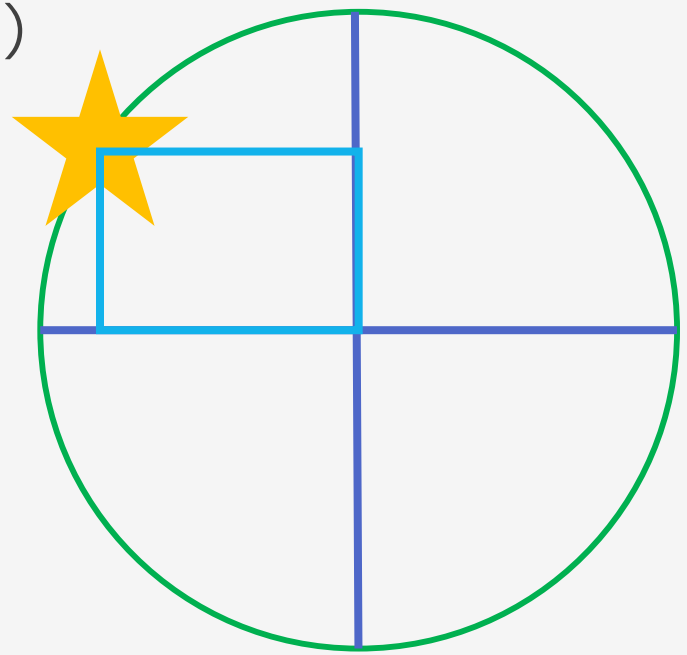
- jCanvasを利用すると少しアニメーションが楽
  - jQuery的な記述が可能
  - 但し、他のオブジェクトと組み合わせて動かすなどが若干面倒

移動

- 移動とはベクトルである
  - ベクトル＝XとY座標における移動量（厳密にはだいぶ違う）
  - どのときに使うか→移動のときに利用する
  - 計算：元のX座標とY座標と移動後のX座標とY座標の差
  - XとY座標の差が移動量となる（これを何秒掛けて移動するか）

直線運動は簡単

- 直線的に動くのであれば、最終地点のX座標、Y座標が分かれば、途中の座標は算出可能
- 問題は曲線、円運動
  - sin、cosを使うことで円運動は表現可能
    - X座標：中心のX座標＋半径＊Math.cos(ラジアン)
    - Y座標：中心のY座標＋半径＊Math.sin(ラジアン)
  - X座標とY座標が決まれば配置が可能



波表現

- Math.sin（ラジアン）はどんな数を入れても-1～1までの値を返す
  - それをY座標に入れれば、波みたいな形を実現することが可能
  - sin波とも呼ぶ

- sin波を利用することで、波みたいな表現が可能
  - 複数を組み合わせると結構キレイ
  - 実際には細かい点を打ちながら直線を繋げて表現する