Table of Contents

Matlab function: Artillery.m

```
% Zweck:
% Rootprogarmm des Matlapprojektes "Spiel Artillery"
% Steuert den Programmablauf
% Ertellt alle benötigten Instanzen
```

Changelog

Hier sind die Commits des ganzen Projektes aufgeführt. Diese wurden aus dem Github übernommen. Die für die jeweilige Klasse Relevanten Versionen sind in der Klasse oim Speziellen aufgeführt. Diese Änderungsliste dient lediglich der Dokumentation des Projektes. Gearbeitet wird mit den Git-Commits:

Commits:

- Version 00.00.01 07.10.15 Raphael Waltenspül Erstellt des Main Objekts, noch nicht Objekt Orientiert.
- Version 00.00.03 15.10.15 Joel Koch Panzer erstellt

- Version 00.00.04 16.10.15 Joel Koch Erstes Gui entworfen
- Version 00.00.05 17.10.15 Joel Koch Lanschaft Algorithmus implementiert
- Version 00.00.06 18.10.15 Joel Koch Powerbar implementiert
- Version 00.00.07 19.10.15 Joel Koch Einschlag implementier
- Version 00.00.08 21.10.15 Raphael Waltenspül Flugparabel berechnung erster entwur
- Version 00.00.09 22.10.15 Raphael Waltenspül Koordinaten der Flugparabel berechnen und ausgeben
- Version 00.00.10 25.10.15 Joel Koch Panzer Detailliert entworfn
- Version 00.00.11 28.10.15 Joel Koch Code Aufgeräumt
- Version 00.00.12 28.10.15 Joel Koch Intersection implementiert
- Version 00.00.13 12.11.15 Raphael Waltenspül Menu zur eingabe von Parametern erstellt
- Version 00.01.00 22.11.15 Raphael Waltenspül Umbau in Objektoriert erfogt
- Version 00.01.01 12.11.15 Raphael Waltenspül Menu zur eingabe von Parametern erweiter auf Objektorientiert
- Version 00.01.02 10.12.15 Raphael Waltenspül Neu Erstellen der Handle Classes GameParameter, Gamestates, Wether
- Version 00.01.03 11.12.15 Raphael Waltenspül Neu Erstellen der Handle Classes Figure, Player
- Version 00.01.04 11.12.15 Raphael Waltenspül Implementieren der Funktionen des Landscapes in Handle Class Landscape
- Version 00.01.05 12.12.15 Raphael Waltenspül Implementieren der Funktione FligthPath in Handle Classes,
- Version 00.01.05 20.12.15 Raphael Waltenspül Implementieren der Funktione Panzer in Player, neu Panzer Farbe, Intersection in Figure,
- Version 00.01.06 28.12.15 Raphael Waltenspül Trefferermittlung
- Version 00.01.07 30.12.15 Raphael Waltenspül Artillery Ablauf
- Version 00.01.08 31.12.15 Raphael Waltenspül Trefferermittlung
- Version 00.01.09 31.12.15 Raphael Waltenspül Implementieren der Funktione Powerbar und Eventhandler in Figure,
- Version 00.01.10 01.01.16 Raphael Waltenspül Neu Entwickeln der Buttons / Game Mode Taktik in Figure
- Version 00.01.11 02.01.16 Raphael Waltenspül Aufräumen fertigstellen Gameablauf
- Version 01.00.00b 03.01.16 Raphael Waltenspül Buglist Testen Kommentieren Dokumentieren

Inputs und Outputs:

Keine

```
% Preconditions:
% Matlab 2015b
```

```
% Postconditions:
% keine
```

Innerhalb der Funktion Artillery zugängliche Variablen

Variabeln

```
% iCount      -- Schleifenzählvariable    (double)
% pk          -- Schleifenzählvariable    (double)
% player()    -- Array welcher alle Player Instanzen enhält ([]Player)
% terrain     -- Lanschaftsarray ([]double)
% tank        -- Tankarray ([double,double])
% windShape   -- windshape handler (objektHandler)
% roundEnd    -- Variable für Programmsteuerung {0= Runde Läft 1=
 Runde Beendet}
% fire        -- Variable für Programmsteuerung {0= Schuss nicht
 erfolgt 1= Schuss erfolgt}
% power       -- Variable für Schussenergie (double)
% angle       -- ariable für Schusswinkel in Grad (double)
% powertimer  -- Timer für Power, wie lange wurde die Maus gedrückt...
 (double)
% coordinate  -- Array mit den flugbahnkoordinaten ([x,y] Double)
% terrain     -- Array mit den Terrainkoordinaten ([x,y] Double) 1002
 length

% Erstellte Instanzen

% param       -- entält alle Benutzereinstellungen    (GameParameter)
% state       -- enthält alle Spielkontrollvariabeln  (GameStates)
% screen      -- Intsatnz des Menu                    (Figure)
% player(x)   -- Instanz eines Spieler                 (Player)
% gameScreen  -- Instanz des Spielfeldes               (Figure)
% lndsc       -- Instanz Landscape erzeugen           (Landscape)
% weth        -- Instanz Wetter                       (Wether)
% shot        -- Instanz der Flugbahn                 (FlightPath)
%
```

Verwendete Instanz Variablen

```
% param.playerQuantety -- Enthält die Anzahl der Mitspieler (double)
% param.numberRounds -- Enthält die anzahl gewünschter spielrunden
 (double)
% param.numberMode -- Enthält Wert für Taktik oder Geschiklichkeit
 {1=T/2=G}
%
% gameScreen.fireEvent -- variable für Programmsteuerung {0= Schuss
 nicht erfolgt 1= Schuss erfolgt}
% gameScreen.mousedown -- variable für Programmsteuerung {0= Maus
 nicht gedrückt1= Maus gedrückt}
% gameScreen.mouseX    -- Xpoition der Maus auf dem Bildschirm
```

```
% gameScreen.mouseY    -- Ypoition der Maus auf dem Bildschirm
%
% player(x).tankHandler -- Enthält das tankhandler objekt
% player(x).livePoints -- Enthält die Lebenspunkte eines Spielers
% player(x).score --  Enthält die Siegespunkte eines Spielers
%
% lndsc.terrainArray -- Enthält das Array der ALndschaft
%
```

Verwendete Instanz Methoden

```
% screen.getFig()      -- gibt den figurehandler zurück (figure)
% screen.drawMenue()   -- zeichnet das Menue ([])
%
% state.getGameRound -- gibt die Aktuelle spielrunde zurück (double)
% state.setPlayerInGame() -- Setzt die anzahl Spielern im Spiel
% state.setActualPlayer() -- Setzt den Spiler welcher am zug ist.
% state.nextPlayer() -- Setzt den Spiler welcher als nächster am zug
 ist
% unter berücksichtigung der aktuellen Lebenpunkte
% state.decreasePlayerInGame -- Reduziert die Anzahl Spieler im Spiel
% gibt die verbleibende Anzahl zurück (double)
% state.getPlayerInGame() -- gibt die Verbleibende Anzahl Spieler
 zurück (double)
% state.nextGameRound() -- nächste Spielrunde, wenn nicht Maximum
% erreicht, sonst game Ende
%
% param.getStandardLivePoints --  gibt die standard Lebenspunkte
 zurück
%
% gameScreen.drawGamescreen() -- Zeichnet das Spielfeld
% gameScreen.drawInScreen() --Landschaft in Spielfeld Zeichnen
% gameScreen.updateInScreen() --Landschaft in Spielfeld Updaten
% gameScreen.drawPlayerPoints() --Zeichnet Die Aktuellen Punktestände
% gameScreen.drawGameRound() --Zeichnet die Aktueller Runde
% gameScreen.drawElementCol() -- [x,y]Array (Bsp. Tank) Zeichnen und
% Handler Speichern Farbe wird Mitgegeben (patchobjekt Handler)
% gameScreen.updateElementCol() --[x,y]Array (Bsp. Wind) Zeichnen,
 altes
% Objekt löschen und Handler Speichern Farbe wird Mitgegeben
 (patchobjekt
% Handler) gameScreen.deleteElement() -- Löscht ein Spielelement
% gameScreen.drawGameButtons() --Zeichnet die Spielbuttons für den
% Taktikmodus gameScreen.drawPowerBar()  -- Zeichnet PowerBar für den
% Geschiklichkeitsmodus gameScreen.drawActualPlayer -- Zeichnet den
 Text
% wer aktuell am Zug ist gameScreen.getPower -- Auslesen der
% Schussenergie Taktik gameScreen.getAngle -- Auslesen des
 Schusswinkels
% Taktik gameScreen.updatePowerBar() -- der Powerbalken wird
 aktualisiert
% gameScreen.drawImpactCircle() -- Zeichnet den Einschlag und gibt das
% Modifizierte Tarrainarray zurück. gameScreen.getFig() -- gibt den
```

```
% Figurehandler zurück
%
% player(x).genTank() -- Tank Berechnen ([])
% player(x).posTank() -- Tank Positioniren ([])
% player(x).getTank() -- Tank Variable ([x,y]Array)
% player(x).getTankColor -- Farbe des Tanks erhalten ([r,g,b])
% player(x).calcAngle -- Berechnet den Winkel in Grad relativ zur
% Mausposition (double)
%
% lndsc.getLandscape()      -- Landschaftarray erhalten ([]double)
% lndsc.genLandscape()      -- Landschaft berechnen ([])
%
% weth.getWindShape         -- Pfeil Shape des Windes
% weth.getWindShapeColor    -- Farbe des Pfeilshapes
%
% shot.calcCoordinates()    --
% shot.isHit()              -- gibt die Trefferenergie 0, 100 zurück
 (double),
% momentan nur 100 spätere Versionen des Spiels können den Tank nur
 beschädigen 0 bis 99.
%
% fig.delete                -- Matlab Methode, löschen einer fig
 instanz
%
```

Verwendete Konstruktoren

```
% GameParameter()
% GameState()
% Figure()
% Player()
% Landscape()
% Wether()
% FlightPath();
%
%
```

Buglist

```
%TODO / Gesamtes Projekt

% #1
% In: coordinate = shot.calcCoordinates(power, angle , weth, lndsc,
 player(state.getActualPlayer));
% Berechnen der Flugbahn
% TODO: try catch entfernen
% das Programm hat noch bugs mit den Arrays Landscape und
% Flighpath zu vergleichen. Dies insbesonder wenn aus dem Bild
% geschossen wird. Es kann ein IndexExceedsMatrixDimensions
% Exeption geworfen werden. Behandelt wird diese momentran noch, in
 dem der nächste Spielr am zug ist...
%
% #2
```

```
% In: comet(coordinate(1,:),coordinate(2,:));
% Zeichnet die Flugbahn des Geschossen
% TODO: Für kurze flugbahnen wird der Schuss sehr Langsam
% gezeichnet.
% Workaround erstellen
%
```

01 Initialisieren des Spieles

löschen des Workspaces

```
function [] = Artillery()clear;

close all;

param = GameParameter;      % Instanz GameParameter erzeugen
state = GameStates;         % Instanz GameStates erzeugen

screen = Figure(state, param);  % Instanz Figure erzeugen
screen.drawMenue();             % Zeichen des Menues
waitfor(screen.getFig());       % Warten bis Menue geschlossen wird
```

02 Spieler Erstellen

in dieser Loop wird ein Array aller Spieler erzeugt

```
% Der Kontruktor Player() Benötigt folgende Variabeln
% iCount - Zälvariable / Nummer des Spielers - (double)
% ['CrashTestDummy', #] - Name des Spieler / Plazhalter für später
% anpassung - (string)
% 'artillery' - Panzertyp / Plazhalter für später variable - (string)
% param - Spielparameter
for iCount = 1 : 1 : param.playerQuantety
    player(iCount) = Player(iCount, ['CrashTestDummy>> ',
 num2str(iCount)], 'artillery', param );
end
```

03 Start des Spieles

in der Folgenden Whileschleife verbleibt das Spiel bis alle SpielRunden (ganzes Spiel) beendet wird.

```
    while state.getGameRound <= param.numberRounds
```

Spielfeld erzeugen

```
gameScreen = Figure(state,param); % Instanz Figure erzeugen
gameScreen.drawGamescreen(); % Spielfeld Zeichnen
```

Spieler Status

Hier wird der Status der aktuell noch im spielbefindlichen Spieler gesetzt. Zu beginn des Spiels auf die Anzahl der eingestellten Mitspieler

```
        state.setPlayerInGame(param.playerQuantety);
```

Landschaft erzeugen und Zeichnen

```
        lndsc = Landscape(param); % Instanz Landscape erzeugen
        lndsc.genLandscape();       % Landschaft berechnen
        terrain =  lndsc.getLandscape(); % Lanschaftsarray erhalten
        gameScreen.drawInScreen(terrain);   % Landschaft in Spielfeld
Zeichnen

        gameScreen.drawPlayerPoints(param, player); % Zeichnet Die
Aktuellen Punktestände
        gameScreen.drawGameRound(param, state); % Zeichnet die
Aktueller Runde
```

Hier wird für jeden Spieler einen Panzer generiert und diesen auf

dem Spielfeld Plaziert und gezeichnet Weiter wird jedem Spieler das GezeichneteObjekt wieder Zurückgegeben, dies um später das Objekt löschen zu können.

```
        for iCount = 1 : 1 : param.playerQuantety
            player(iCount).genTank; % Tank Berechnen
            player(iCount).posTank(lndsc, param); % Tank Positioniren
            tank = player(iCount).getTank; % Tank Variable
            % Tank Zeichnen und Handler Speichern
            player(iCount).tankHandler =
gameScreen.drawElementCol(tank,player(iCount).getTankColor);
        end
```

Wetter Erstellen

```
        weth = Wether(param); %Wetter Instanz erstellen
        windShape =
gameScreen.drawElementCol(weth.getWindShape,weth.getWindShapeColor); %Wetterpfeil
Zeichnen
```

Zug ermitteln

Hier wird ausgelost welcher Spielr anfängt

```
        state.setActualPlayer(round((rand) *
(param.playerQuantety-1))+1);
```

Taktik oder Geschicklichkeit

Liest aus den Parametern ob der Taktik oder Geschiklichkeitsmodus eingestellt wurde. Ladet den Mode und Zeigt diesen an.

```
        if param.numberMode == 1;
```

```
            gameScreen.drawGameButtons(); % Zeichnet Buttons für den
Taktikmodus
        else
            gameScreen.drawPowerBar();  % Zeichnet PowerBar für den
Geschiklichkeitsmodus
        end
```

Start einer Spielrunde

In dieser While Schleife verbleibt das Spiel, bis eine Runde beendet wird.

```
        roundEnd = 0; % Variable zur Spielsteurung
        while roundEnd == 0;
```

Wetter Ermitteln

Hier wird das Wetter neu berechnet, der wind ändert sich leicht (normalverteilter zufall)

```
            weth.updateWether; % neu berechnen
            windShape = gameScreen.updateElementCol(windShape,
weth.getWindShape,weth.getWindShapeColor); %windShape neu zeichnen

            gameScreen.drawActualPlayer(state,
player(state.getActualPlayer).getTankColor); %Zeichnet den Text wer
aktuell am Zug ist
```

Warten auf Eingabe

Nachfolgend wird auf die eingabe des Spielers gewartet

```
            fire = 0; % Wurd geschossen 1 wenn schuss erfolgt
            power = 1000; % Schussnergie (sollte zwichen 1000 und
100000 Joul liegen)
            angle = 45; % Abschusswinkel in Grad
            % Ermitteln ob Taktik oder Geschiklichkeit
            if param.numberMode == 1;
```

Taktikmodus

warten auf Feuerbefehl Taktik

```
                while gameScreen.fireEvent == 0;
                    pause(0.1)
                end
                gameScreen.fireEvent = 0; % Feuerbefehl wieder auf
Null setzen Taktik
                power = gameScreen.getPower; % Auslesen der
Schussenergie Taktik
                angle = gameScreen.getAngle; % Auslesen des
Schusswinkels Taktik

            else
```

Geschiklichkeitsmodus

Warten auf eingabe Geschiklichkeit Polling schleife. Falls Mouse down, zählt die Powerbar nach oben

```
                powertimer = 0; % Timer für Power auf null setze
                while fire == 0 % Warten bis eingabe erfolgt
                     pause(0.01); % Pause, wird benötigt um elemente zu
Zeichnen.
                     while gameScreen.mousedown == 1  % Wenn maus
gedrückt wird fängt das Hochzälen an
                          powertimer = powertimer * 1.005 + 0.25; % Der
Timer zählt hoch und beschleunigt zusätzlich
                          gameScreen.updatePowerBar(powertimer/180);   %
der Powerbalken wird aktualisiert
                          fire = 1; % Der Feuerbefehl wird auf erfolgt
gesetzt
                          angle =
player(state.getActualPlayer).calcAngle(gameScreen.mouseX,gameScreen.mouseY); %De
Winkel Spieler zus mausposition wird ausgelesen.
                          power = powertimer * 200; %der Poerwert wird
berechnet
                          pause(0.000001);  % pause um geschwindikeit des
Hochzählens zu beschränken Ansonsten abhängik von rechner Leistung
                     end
                end

            end
```

Schuss Rechnen

Hier Wird der Schuss berechnet

```
            shot = FlightPath(); % Instanz des Schusses erstellen
            % Wenn Spielernummer Gerade, wird der Abschusswinkel
gespiegelt
            if mod(state.getActualPlayer, 2) == 0
                angle = 180 - angle;
            end

            % Berechnen der Flugbahn
            % TODO: #1 try catch entfernen
            % das Programm hat noch bugs mit den Arrays Landscape und
            % Flighpath zu vergleichen. Dies insbesonder wenn aus dem
Bild
            % geschossen wird. Es kann ein
IndexExceedsMatrixDimensions
            % Exeption geworfen werden. Behandelt wird diese momentran
noch, in dem der nächste Spielr am zug ist...
            % Joel Koch 4.1.16: Problem behoben.

            try
                coordinate = shot.calcCoordinates(power, angle , weth,
lndsc, player(state.getActualPlayer));
```

```
            catch
            end
```

Schuss Zeichnen

Zeichnet die Flugbahn des Geschossen TODO: #2 Für kurze flugbahnen wird der Schuss sehr Langsam gezeichnet. Workaround erstellen comet(coordinate(1,:),coordinate(2,:));

```
            tmpX= (coordinate(1,:));
            tmpY= (coordinate(2,:));
            fprintf('\nDebug Info Arraysize Comet Coord = %d \n',
 length(tmpX));
            comet(tmpX,tmpY);
```

Treffer ermitteln

Hier Wird ermittelt ob der einschlagpunkt auf einem der Spieler ist

```
            for pk = 1 : 1 : param.playerQuantety % Für jeden Spieler
                try % Siehe TODO #1
                    if shot.isHit(player, pk) > 0   % Die
 Trefferenergie 0, 100 wird ausgelesen, spätere Versionen des Spiels
 können den Tank nur beschädigen.

 gameScreen.deleteElement(player(pk).tankHandler); % wenn Getroffen
 wird Spielfigur gelöscht
                        player(pk).livePoints = player(pk).livePoints
 - 100;   % Die Lebenspunkte des Spielers werden auf 0 gesetzt
                        player(pk).positionXY = [0,0]; % player kann
 nicht mehr getroffen werden
                        if state.decreasePlayerInGame <= 1  % die
 Anzahl verbleibenden Spieler wird reduziert
                            roundEnd = 1; % Ist die Anzahl kleiner 1,
 so endet die runde
                        end
                    end
                catch
                end
            end
```

Einschlag Schuss Zeichnen

Hier wird die Explosion gezeichnet und das neue Terrainarray mit Krater gespeichert

```
            try  % Siehe TODO #1
                lndsc.terrainArray =
 gameScreen.drawImpactCircle(lndsc.getLandscape, shot.impact);
            catch
            end

            terrain =  lndsc.getLandscape(); % lokalspeichern des
 Terrainarray
```

```
            gameScreen.updateInScreen(terrain); % neuzeichnen des
 Terrainarray
```

Ermitteln des Nächsten Spieler

Nächster Spieler mit mehr als 0 Lebenspunkten wird ermittrelt Ist nur noch ein spieler vorhanden Erhält der verbleibende einen Punkt

```
            if state.getPlayerInGame > 1
                state.nextPlayer(param, player);
            else
                state.nextPlayer(param, player);
                %Punkt für den letzten verbleibenden Spieler vergeben
                player(state.getActualPlayer).score =
 player(state.getActualPlayer).score+1;
            end
            pause(0.1); %Pause zum ende der While Schleife, für
 Stabilität eingabe etc.

        end
        state.nextGameRound(param); %Runde eins hochzählen wenn max
 anzahl erreicht ende

        for iCount = 1 : 1 : param.playerQuantety % rücksetzen aller
 lebenspunkte
            player(iCount).livePoints = param.getStandardLivePoints;
        end

        % löschen des Screens wird anschliessend neu gezeichnet
        if state.getGameRound > param.numberRounds
        pause(13);
        end
        fig = gameScreen.getFig();
        fig.delete;

    end

end
```

Published with MATLAB® R2015b