Comp 388/488 - Game Design and Development

Spring Semester 2018 - Slides - week 13

Dr Nick Hayward

Final Presentation & Report

- team presentation on Friday 27th April 2018 @ 2.45pm
- team report due on Friday 4th May 2018 by 2.45pm

Final Assessment Outline

- final design of game from DEV Week…
    - *continue to develop your group's game concept*
- working game (as close as possible)
- explain choices made in the design and development
    - *initial choices*
    - *final implementation choices, options, patterns…*
- show and explain implemented differences from DEV Week
    - *where and why did you update the game?*
    - *how did playtesting influence your updates and designs?*
    - *perceived benefits of the updates?*
- how did you respond to feedback from DEV Week and onwards…
- anything else you consider relevant to your game
    - *within reason...*

n.b. ~ 10 minutes per group

Final Assessment Report

- report outline - demo and report

Project Status Report

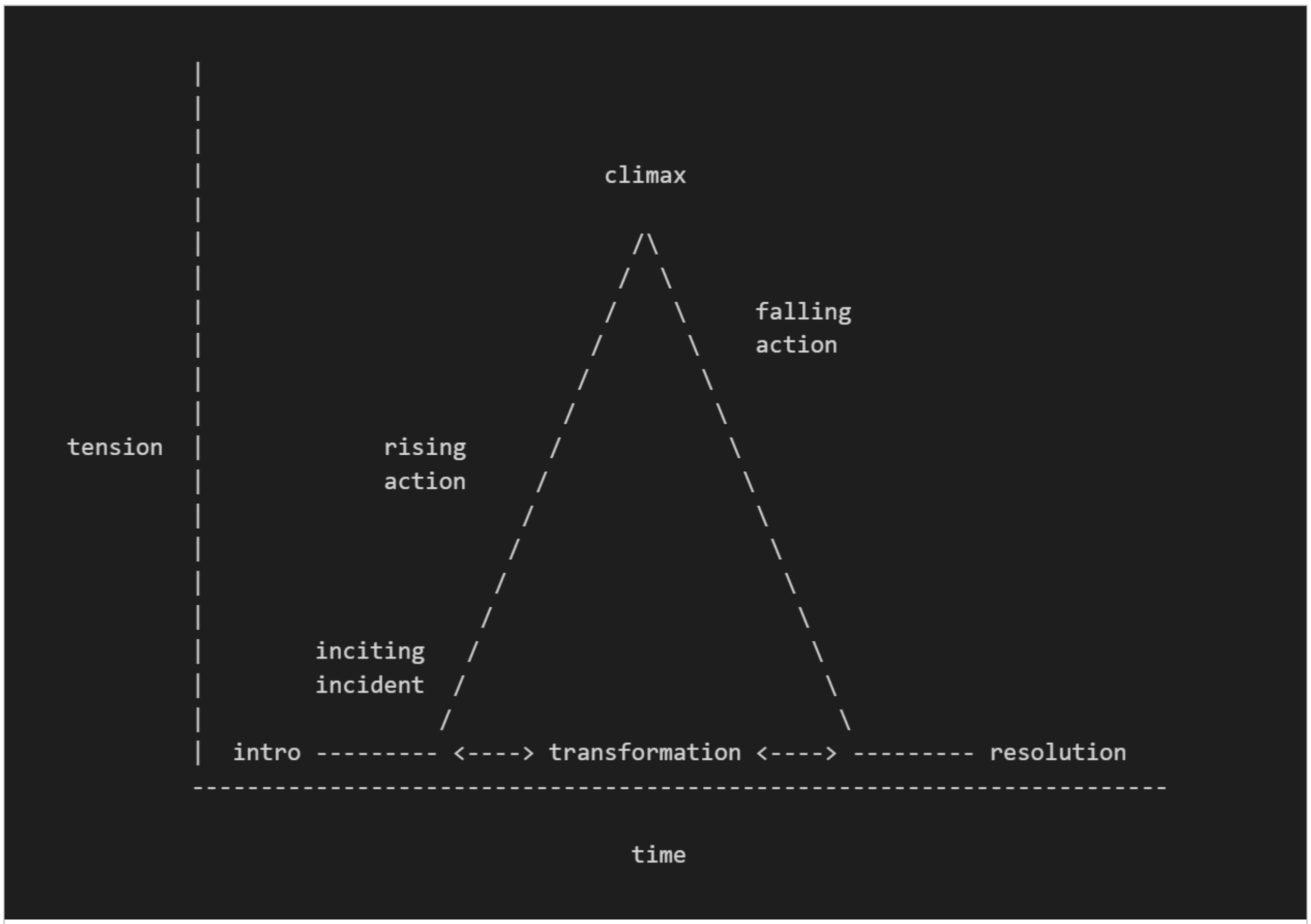
- what is currently working?
  - *what is currently playable?*
- what is left to add or fix?
  - *features, characters, challenges &c.*
- who is working on what?
  - *logic, design, testing, research...*
- any special requirements for demo and presentation?

Teams

- Group A - Team Shinde wa Ikenai
- Group B - Team Split
- Group C - Team You Snooze You Lose
- Group D - Team Lords of the Arena
- Group E - Team Venturer
- Group F - Team BlockRobot
- Group H - Team N/A
- Group I - Team Swordfish
- Group J - Team Tristram
- Group K - Team Haunted House

Image - Narrative Structure

conflict in a game's story



```
          |
          |
          |
          |                             climax
          |
          |                               /\
          |                              /  \
          |                             /    \     falling
          |                            /      \    action
          |                           /        \
          |                          /          \
tension   |             rising      /            \
          |             action     /              \
          |                       /                \
          |                      /                  \
          |                     /                    \
          |                    /                      \
          |        inciting   /                        \
          |        incident  /                          \
          |                 /                            \
          |  intro --------- <----> transformation <----> --------- resolution
          -----------------------------------------------------------------------

                                  time
```

Narrative Structure

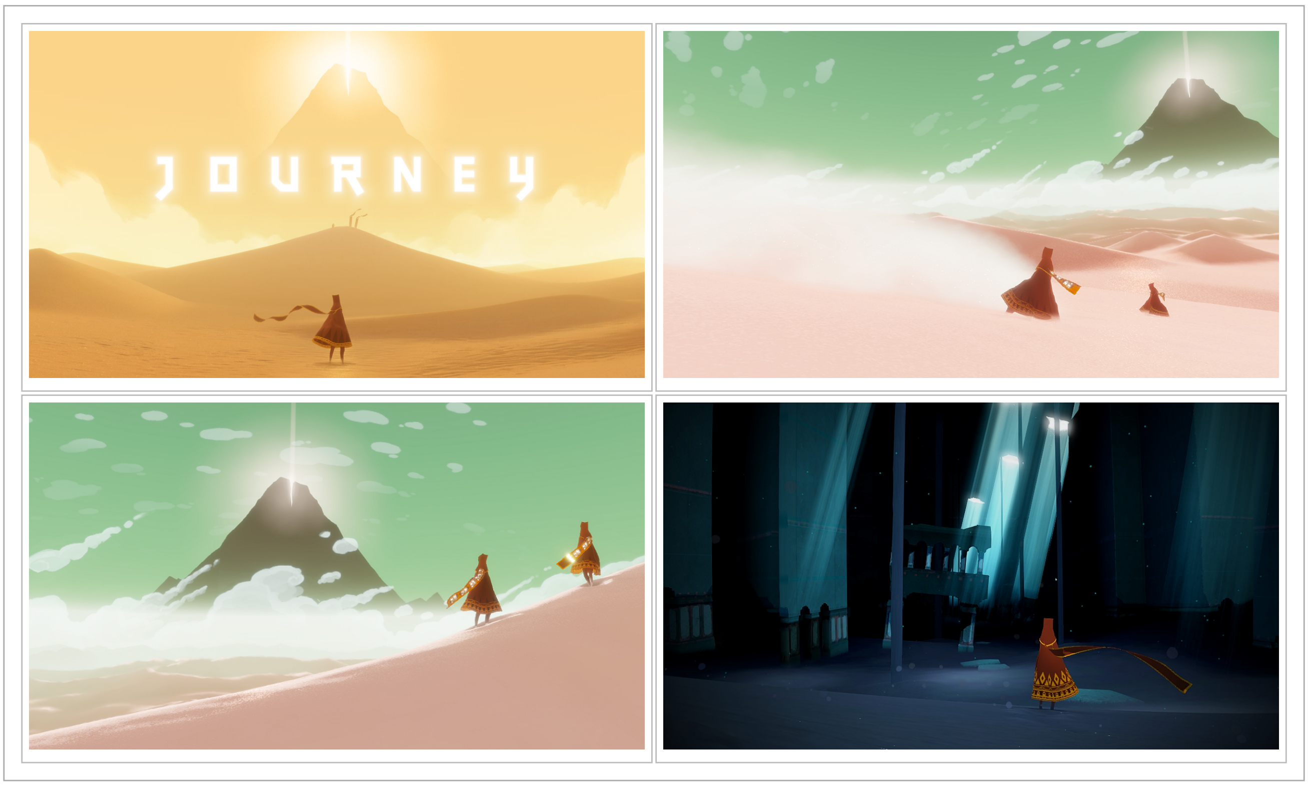
- Source - Building An Arc: Bringing Narrative Structure To Your DJ Sets

Games and dramatic elements

Journey to a narrative structure

- a recent example of narrative structure in gaming was the 2012 release *Journey*.
  - *designed by ThatGameCompany, and directed by Jenova Chen*
- its underlying design and story was inspired by *The Hero's Journey*
  - *a structure and outline for myth and story telling prescribed by Joseph Campbell*
- Campbell defined twelve stages on the *Hero's Journey*
  - *set a structure that follows the narrative arc along the path of the story*
- initial incident is an effective acknowledgement of the limits of the current environment
  - *the encompassing world for the hero*
- the hero must now leave this environment, this comfort zone of sorts
  - *embark into unknown, commonly dangerous territory*
- this journey will normally include many trials and tests
  - *the challenges we expect to introduce to many games*
- trials are not simply physical, but may also include
  - *aspects of temptation, mental reasoning, emotional dilemmas...*
- player will normally be expected to reach a defined low point on the journey
  - *the abyss that defines and shapes the counterpoint to the story and game*
- introduction of an extreme low point, the *abyss*
  - *allows the character to metaphorically die*
  - *then be reborn ready for the final challenges of the journey*
- the hero will then return to a point of calm and resolution
  - *transformed and free of the issues, fear, and doubts that initially defined them...*

Image - Journey



- Source - ThatGameCompany

Video - Journey

available on PS4

- Source - Journey PlayStation 4 Official Launch Trailer - YouTube

Week 12 reminder - quick exercise

Consider the following game characters and objects,

- a medieval knight
  - *carries a sword, may ride a horse, fighting skills, finite health...*
- a squire
  - *attends to the knight*
- a semi-intelligent/aware mob object - e.g. an ogre
  - *carries a club, may ride horse-like animal, fighting skills, renewable health...*
- a series of huts, caves &c. in the gaming world

Each of these characters or objects may be pre-defined or created with a sense of free will.

Define the following,

- rules for each character and object
- a brief outline for a game with these characters and objects

Then consider the following,

- how might free will affect the rules and outline for your initial game?
- what type of unexpected glitches, interactions, and features may result due to free will in this game?

Week 13 update - quick exercise

For the earlier game, characters, and objects you defined, consider the following

- reduce this game's outline to its bare essentials
  - *i.e. which shapes, patterns, colours, objects &c. are still necessary to define your game's story?*
- outline the narrative structure for this game using these bare essentials
  - *where is the conflict in this story? the rising action, climax, falling action &c...*
  - *how is the resolution achieved for this game's narrative structure?*

Then, re-consider the role and influence of free will or emergent systems on this narrative structure

- what type of unintentional features, dead ends &c. may be introduced?
- how do you allow for such potential issues in your narrative structure?

Python and Pygame - Game Example I

fun game extras - load explosion images

- need to be able to define and load our images for the explosions
  - *use a list for these images*
  - *then cycle through these explosions as required...*
- our first exanple will use a list to simply load these explosion images
  - *initially use a `for` loop to iterate over this directory and load our images, e.g.*

```
# explosions
explosion_imgs = []

# iterate over explosion images in directory
for i in range(9):
    file = 'explosion{}.png'.format(i)
    expl_img = pygame.image.load(os.path.join(img_dir, file)).convert()
    expl_img.set_colorkey(BLACK)
    explosion_imgs.append(expl_img)
```

- use built-in function, `format()`, to specify abstracted value for iterator index
  - *in this example abstracted for the required filename*
- create our image for the Pygame window
  - *set colour key to black to create our transparency for the containing shape's background*
- then append these images to our list for explosions

Python and Pygame - Game Example I

fun game extras - create explosion sprite object - part I

- create a new class to help us represent and organise our sprite object for *explosions*
- add a new class for this object
  - *then start by initialising this sprite, e.g.*

```python
# create a generic explosion sprite - use for asteroids, player explosions &c.
class Explosion(pygame.sprite.Sprite):
    # initialise sprite
    def __init__(self, center):
        pygame.sprite.Sprite.__init__(self)
        ...
```

- after initialising this new sprite object
  - *set starting image for our explosions*
  - *set to first index position of our list for explosion images*
- need to add the rectangle for this image
  - *set its centre to the specified value of the passed parameter*
- also set initial frame for our animation
  - *we can set it to a starting default of 0*

Python and Pygame - Game Example I

fun game extras - create explosion sprite object - part 2

- animation needs to be steady and constant
  - *may create a steady framerate for the animation itself*
  - *now check the time in ticks for the last update*
- then set a default framerate for this animation
  - *modify framerate of animation to suit game requirements*

```python
# create a generic explosion sprite - use for asteroids, player explosions &c.
class Explosion(pygame.sprite.Sprite):
    # initialise sprite
    def __init__(self, center):
        pygame.sprite.Sprite.__init__(self)
        # specify image for explosion sprite
        self.image = explosion_imgs[0]
        # set rect for image
        self.rect = self.image.get_rect()
        self.rect.center = center
        # set initial frame for animation
        self.frame = 0
        # check last update to animation
        self.last_update = pygame.time.get_ticks()
        # set framerate delay between animation frames - sets speed for explosion
        self.frame_rate = 50
```

Python and Pygame - Game Example 1

fun game extras - create explosion sprite object - part 3

- need to add an `update` function to our class
  - *updates image of explosion for this sprite object as time progresses*
  - *i.e. as the framerate advances, switch explosion images to create animation*

```
...
# change image as time progresses for explosion sprite
def update(self):
    # get current time
    now = pygame.time.get_ticks()
    # check if enough time has passed between animations
    if now - self.last_update > self.frame_rate:
        self.last_update = now
        # if enough time passed - add 1 to frame
        self.frame += 1
        # check if end of explosion images reached
        if self.frame == len(explosion_imgs):
            # kill if end of image reached
            self.kill()
        else:
            center = self.rect.center
            self.image = explosion_imgs[self.frame]
            # update rect for image
            self.rect = self.image.get_rect()
            self.rect.center = center
```

- need to check the current time in the game
  - *check if enough time has passed between each animation*
- if enough time has elapsed
  - *update the value for the `last_update` time record*
  - *advance our animation frame by an increment of 1*
- then `kill()` animation at the end of the explosion images...

Python and Pygame - Game Example 1

fun game extras - add explosions to collisions

- our sprite object for explosions has now been created
- now call this explosion whenever we record a collision between
  - *a projectile and a mob object*
  - *a mob object and player object...*
- in our game loop `update` section
  - *check for collisions we can now add an animation for the explosions*

```python
...
# add more mobs for those hit and deleted by projectiles
for collision in collisions:
    # calculate points relative to size of mob object
    game_score += 40 - collision.radius
    # play explosion sound effect for collision
    explosion_effect.play()
    # add animation for explosion images if collision
    explosion = Explosion(collision.rect.center)
    # add explosion sprite to game sprites group
    game_sprites.add(explosion)
    # create a new mob object
    createMob()
...
```

- as we're checking for collisions, we can now
  - *update game score*
  - *play a sound effect for an explosions/collision*
  - *create the animation for the explosion effect*
  - *...*

resources

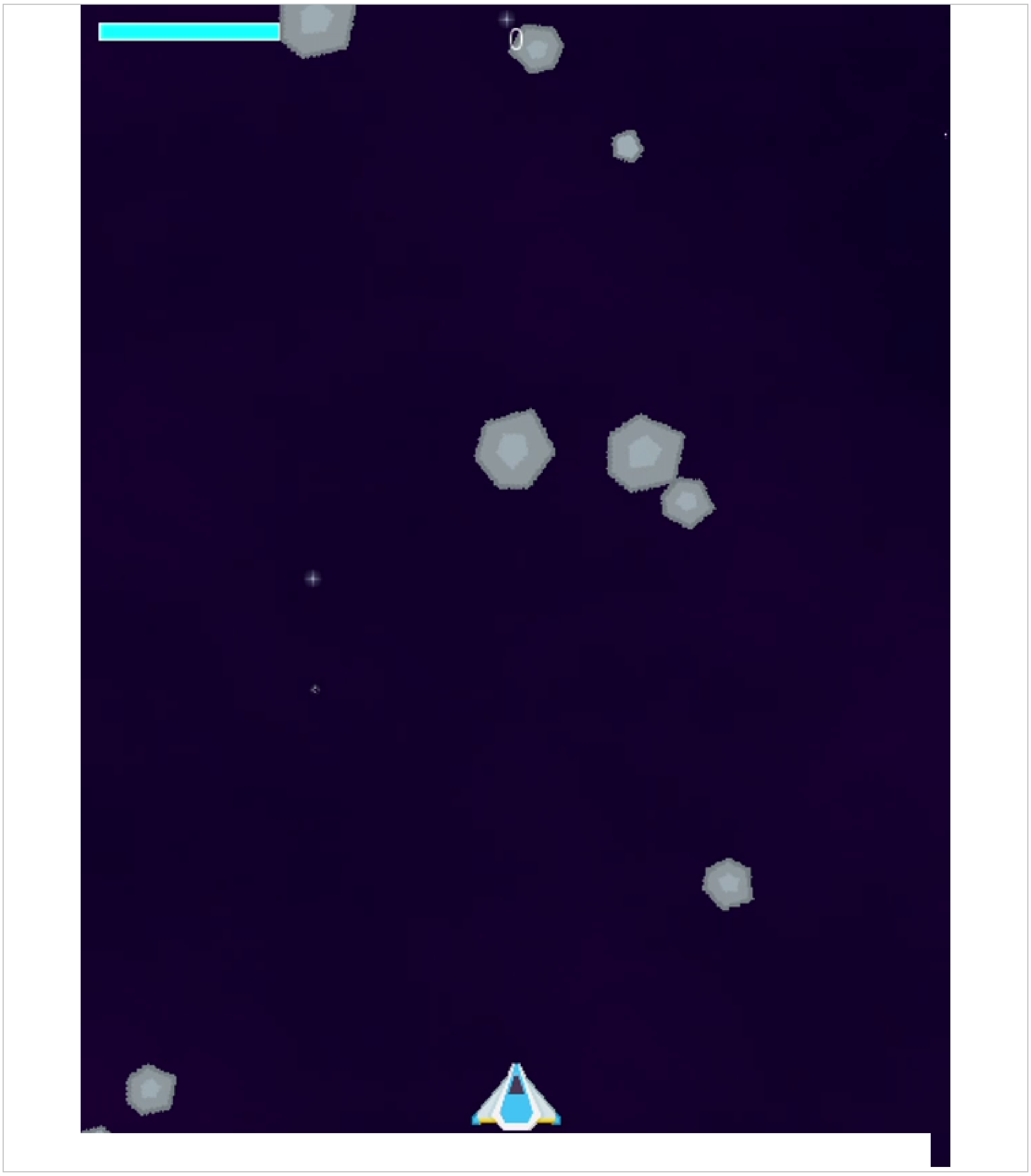
- notes = extras-part1-explosions.pdf
- code = objectexplosions1.py

game example

- shooter1.2.py
- add some fun explosions
  - *create sprite object for explosion*
  - *cycle through images to create explosion animation*
  - *add explosion for each collision*

Video - Shooter 1.2 - Part 1

add some fun explosions - mob objects



Game designers

Designer example - Rob Pardo

- Pardo is best known as the lead designer of Blizzard's **World of Warcraft**
- various jobs and titles at Blizzard, including
  - *lead designer and Executive Vice President of Game Design*
  - *before becoming Chief Creative Officer until the middle of 2014*
- his best known games include, for example,
  - *World of Warcraft*
  - *WarCraft*
  - *StarCraft - now free to download*
  - *Diablo*
  - *Mortal Kombat*
- Pardo was instrumental in pushing a different concept for WarCraft
  - *more towards what we now consider traditional RPG games...*
- with the introduction of 3D for WarCraft III
  - *they tested various options for camera usage in this type of game*
- after experimenting with different angles and perspectives
  - *including a lower shooting position*
  - *they settled on the now familiar, traditional isometric view*
- assessment of camera options became a key factor in this game's development
  - *informing many of the early 3D prototypes for this game*
- prototyping also allowed Pardo and his design team
  - *to iteratively determine the nature of units and heroes in the game*
- such concepts and designs helped shape the nature of the game
  - *its story, possible objectives, characters, units...*
  - *e.g. the development of WarCraft's races*
- MIT Game Design lecture - https://youtu.be/jVzgWmNJlY4

Video - World of Warcraft

Rob Pardo on WarCraft III

- Source - YouTube

Games - Complete and Functional

check initial functionality

- eventually you'll need to allow testers and players to actually play the game
- there may be rough edges to your game
  - *perhaps even broken code in places*
  - *but you should be working towards a functional game...*
- e.g. some form of a test version of your game
  - *that allows an unfamiliar player to experience and play your game*
- a lack of guidance and prior knowledge of the game's design and development
  - *often helps in this determination of functionality*
- test players should be able to interact with your game
  - *interact without your influence or interference*
  - *might include paper prototypes or an early example of the digital game itself*
- how do we determine if a game is functional?
  - *often a matter of subjective judgement*
- a useful heuristic to determine a functional game
  - *consider if a test player can complete a session without a developer's guidance and advice*
- after determining the functional nature of a game
  - *begin a consideration of a game's functionality compared to completeness*
- even if a game is functional, there will still be aspects that are incomplete
  - *including unintended loopholes, dead ends, glitches...*
  - *some may be useful, others popular with test players &c.*

Games - Complete and Functional

check completeness

- test every aspect of a game to ensure completeness
- checking for incomplete parts of our game
- might include issues with a game's logic, rendering, performance...
  - *or a result of poorly defined rules and procedures*
- for an identified incomplete game section
  - *initially begin by considering the defined rules for the game*
- checking the game's design document and prescribed rules
  - *ensure there are no mistakes, contradictions, or gaping holes in the conceptual logic*
- after fixing an identified issue
  - *also need to check for any knock-on effects with other parts of our game*
- need a few testing sessions and checks
  - *helps ensure completeness has been initially considered and resolved*
  - *often becomes a rolling series of checks, updates, and re-testing*
- also determine issues as play testers become involved in the process
  - *detect issues that are not necessarily a result of ambiguous rules*
  - *perhaps a result of unintentional gaming options*
- well-known example of this unintentional feature or issue is the *spawn camping* problem. e.g.
  - *Camping in Games*
  - *seen in FPS games, such as Call of Duty and Rainbow Six Siege*
  - *not unique to FPS...*

Video - Call of Duty 4

Extreme Spawn Camping

- Source - YouTube

Games and development

quick exercise - part 1

Choose at least one of the following games,

- Asheron's Call
- Asteroids
- Deus Ex
- Journey
- Mario Kart

or use your own game idea and concept.

Then consider the following questions:

- what is the minimum you consider necessary for this game to be functionally complete?
  - *in effect, ready for initial testers and players?*
- as a designer and developer, which aspects of the game would you leave open to change during testing?

Python and Pygame - Game Example I

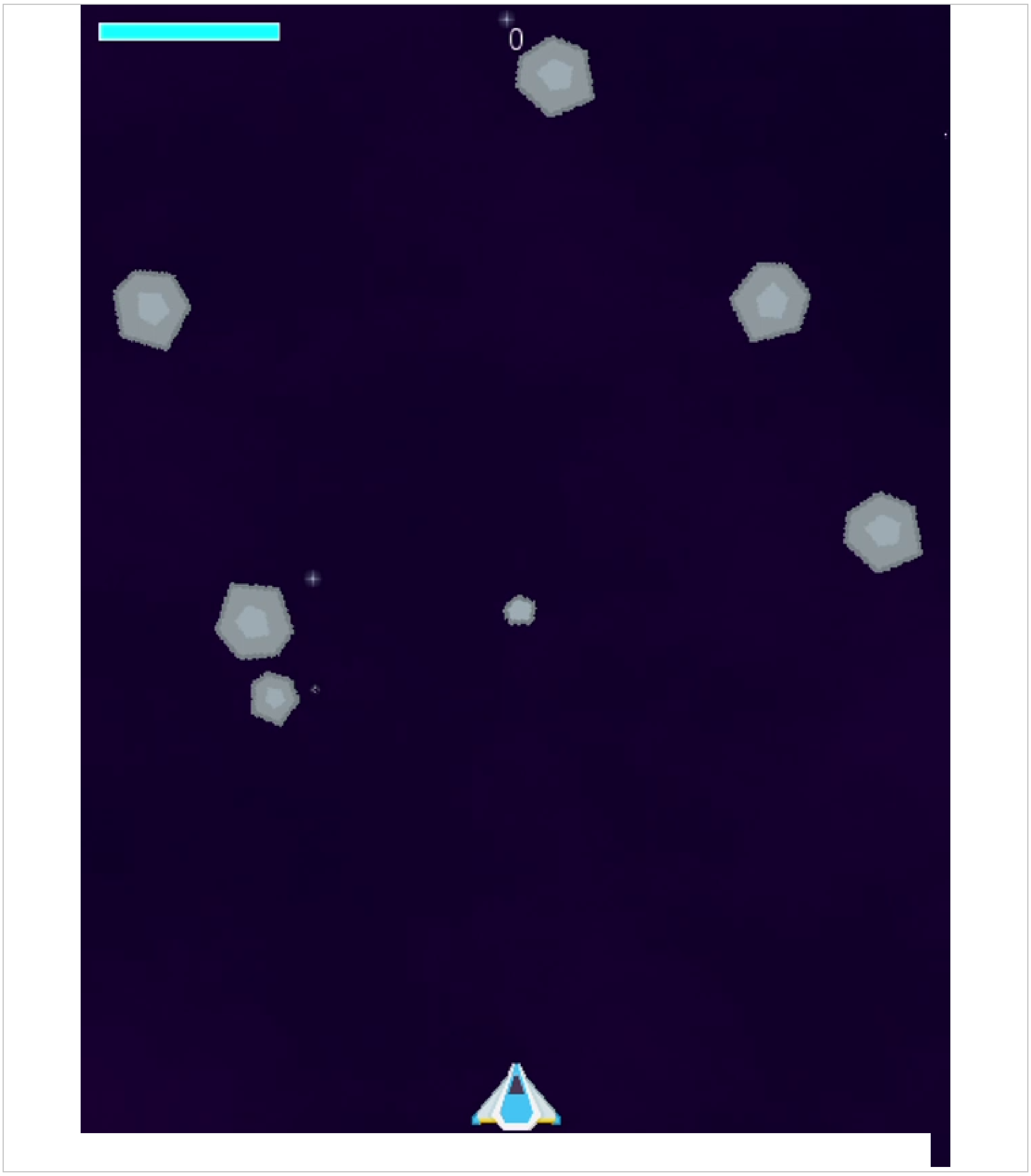
fun game extras - add explosions to player's ship

- add explosions to a collision with a player's ship
  - *again, update game loop for these collisions*

```
# add check for collision - enemy and player sprites (True = hit object is now deleted from game window)
collisions = pygame.sprite.spritecollide(player, mob_sprites, True, pygame.sprite.collide_circle)
# check collisions with player's ship - decrease shield for each hit
for collision in collisions:
    # decrease player's shield for each collision
    player.stShield -= 20
    # add animation for explosion images if collision
    explosion = Explosion(collision.rect.center)
    # add explosion sprite to game sprites group
    game_sprites.add(explosion)
    # create a new mob object
    createMob()
    # check overall shield value - quit game if no shield
    if player.stShield <= 0:
        running = False
```

Video - Shooter 1.2 - Part 2

add some fun explosions - player's ship



Resources

Demos

- pygame - fun game extras
- objectexplosions1.py
- pygame - Game 1 Example
- shooter1.2.py

Games

- Call of Duty
- Command & Conquer
- Journey - ThatGameCompany
- Journey - PS3
- Journey - Wikipedia
- Rainbow Six Siege
- Space Quest
- StarCraft
- free download
- World of Warcraft

Game notes

- Pygame
- extras-part1-explosions.pdf

References

- Bogost, I. *Persuasive Games: The Expressive Power of Videogames*. MIT Press. Cambridge, MA. 2007.
- Bogost, I, *The Rhetoric of Video Games*. in *The Ecology of Games*... Salen, E. MIT Press. Cambridge, MA. 2008.
- Bogost, I. *Unit Operations: An Approach to Videogame Criticism*. MIT Press. Cambridge, MA. 2006.
- Csikszentmihalyi, M. *Flow: The Psychology of Optimal Experience*. Harper & Row. New York. 1990.
- Huizinga, J. *Homo Ludens: A Study of the Play-Element in Culture*. Angelico Press. 2016.
- Murray, J. *Hamlet on the Holodeck: The Future of Narrative in Cyberspace*. Free Press. New York. 1997.
- Poundstone, W. *Prisoner's Dilemma*. Touchstone. New York. 2002.
- Salen, K. & Zimmerman, E. *Rules of Play: Game Design Fundamentals*. MIT Press. 2003.
- Various
- ThatGameCompany - Hiring

Videos

- Extreme Spawn Camping
- Journey PlayStation 4 Official Launch Trailer - YouTube
- Rob Pardo on Warcraft III