Comp 388/488 - Game Design and Development

Spring Semester 2018 - Notes - week 5

Dr Nick Hayward

Games and engagement - learning to play again

concept & premise

- formal structures for a game may also benefit from a concept or premise
  - *helps frame or wrap the general gameplay*
- we're creating an underlying reason for a given game
- something we hope our players will enjoy
  - *and consider worthy of their time and investment*
- a concept or premise is a great way to hook our players into a game
- player needs a valid reason to play the game
  - *rarely just the mechanics...*
- **Space Invaders** has a simple hook for the game and play
- for **Mario** games, Miyamoto uses a simple premise to wrap the mechanics and gameplay
  - *progress through varied levels to save the Princess*
  - *a means of showcasing each game's formal structures*

Games and engagement - learning to play again

story and characters

- development of video games includes a shift in design and story telling
  - *e.g. towards a consideration of characters*
- character development has been growing since the earliest games
  - *a useful, fun part of developing a premise for a game*
  - *Mario, Donkey Kong, Sonic, Pac Man...*
- each character development acted as a tool to help engage players
- characters help a player to become engaged and immersed
  - *in the general premise of a game*
  - *a specific story in particular*
- characters act as a direct link and interaction
  - *between a game's narrative and its player*
- designers and developers may also use characters
  - *a means to manipulate stories, and general gameplay...*
- a player may impart their own characteristics and personality on a game character
  - *need to be careful not to restrict a character too much*
- players often deeply invested in a game due to characters
  - *they form an attachment with characters...*
  - *conventions and cosplay continue to grow in popularity*
- story and characters may fulfill a dramatic context within our game
  - *depends how far we wish to push such elements within our game*

Games and development

Enter the Mummy's Tomb - objects, attributes...

- in our earlier game, *Enter the Mummy's Tomb*, we introduced three initial characters
  - *explorer (our Egyptologist)*
  - *high priest*
  - *scary pharaoh*
  - *the mummy*
- objects may include known characteristics and attributes from real world, e.g.
  - *name*
  - *health*
  - *current value & status, lives, regeneration...*
  - *physical characteristics*
  - *height, speed, strength, vision...*
  - *skills*
  - *fighting, shooting, intelligence (problem solving &c.) ...*
  - *motion*
  - *e.g. walking, running...*
  - *actions*
  - *pick-up, throw, move, drop...*
- each character possesses such attributes, to a greater or lesser extent...
- may also reuse such attributes as a definition, template
  - *help guide the subsequent development of other characters*
- new characters might include
  - *earth-bound creatures*
    - horse, scarab beetle, snake...
  - *Egyptian gods*
  - *e.g. Anubis, Osiris, Isis, Horus, Ma'at, Sekhmet, Seth...*
  - *enemies*
  - *allies...*
  - *other explorers...*

Games and development

Enter the Mummy's Tomb - attributes...

- consider attributes useful and applicable to each of our main characters
  - *characteristics and actions our characters may need and use in the game*

explorer	high priest	scary pharaoh/mummy
name	name	name
health	health	health
fight/attack		fight/attack
	help/aid	
info	info	info
retreat		retreat

- list of attributes is not exhaustive, and it may grow as we develop a game
- may also find it useful to combine some of these attributes into a given class
  - *fight and health attributes may only apply for an **attack** method*
- may also consider the tombs as an additional object within our game
  - *attributes may include, e.g.*

tomb
name/number (e.g. KV17)
owner
owner type
find treasure
info

- may start to see common attributes and characteristics
- create methods to help us structure and call such characteristics within a given class
  - *e.g. a class for the explorer*
- *owner* of each tomb is unknown until we randomly pick a character
  - *may be an instance of the high priest or the mummy class*
- *owner type* may end up either helping or attacking the *explorer*

Games and development

Enter the Mummy's Tomb - initial structure

- many of these objects share common traits and attributes
  - *explorer, high priest, and mummy may use inheritance*
- allows us to create a useful superclass/parent class
  - *this will be our initial* ***GameCore***
- *GameCore* may include the following:
  - *attributes*
    - name
    - owner
    - health
  - *methods*
    - fight/attack
    - help/aid
    - info
    - retreat
- add to the *GameCore* as we build out our current game
  - *each of the characters may inherit from this GameCore class*
  - *each character class may also override default methods*
- for example
  - *give the explorer enhanced options to fight/attack*
  - *perhaps the mummy will have a higher initial health value*
- a *tomb* may also inherit certain default attributes from the *GameCore*
  - *including name and info*
- each *tomb* will also contain, or be composed of, another object
  - *such objects may be used to perform specific tasks*
  - *perhaps an owner composed of a high priest or mummy*

quick exercise

consider the following 4 characters:

- poet / bard
- archer
- scout
- knight

then outline the following:

- abstract objects and attributes for all of these characters
- show developer pattern from abstract to specific character
- show relationship between character objects and attributes
- similarities and differences between developer and player updates
  - *for abstract and specific characters...*

Games and engagement - learning to play again

pushing boundaries

- many examples we may reference as archetypal games
- many games break this mould
  - *may even push the standard perception of a game*
- recent development in games is towards the use of immersive environments
  - *simply promote calm and relaxation.*
- gaming to reduce stress by exploration
  - *instead of high paced action and adventure*
- a natural progression from earlier games, e.g. *Civilization*, *Age of Empires*...
  - *to a new audience and emerging genres*
  - *Abzu, Journey, Proteus...*
- annual *Games for Change* festival in New York
  - *considers games in a broader social context*
  - *Games for Change*
- boundaries are also being pushed with indie development and experimental gaming concepts
- *Independent Games Festival*
  - *a great place to start exploring such ideas and concepts*
  - *Independent Games Festival*
- *IndieCade* festival, the International Festival of Independent Games
  - *IndieCade*

Video - IndieCade 10th Anniversary

Source - IndieCade - YouTube

Video - Abzu

Source - Abzu trailer - YouTube

Game designers

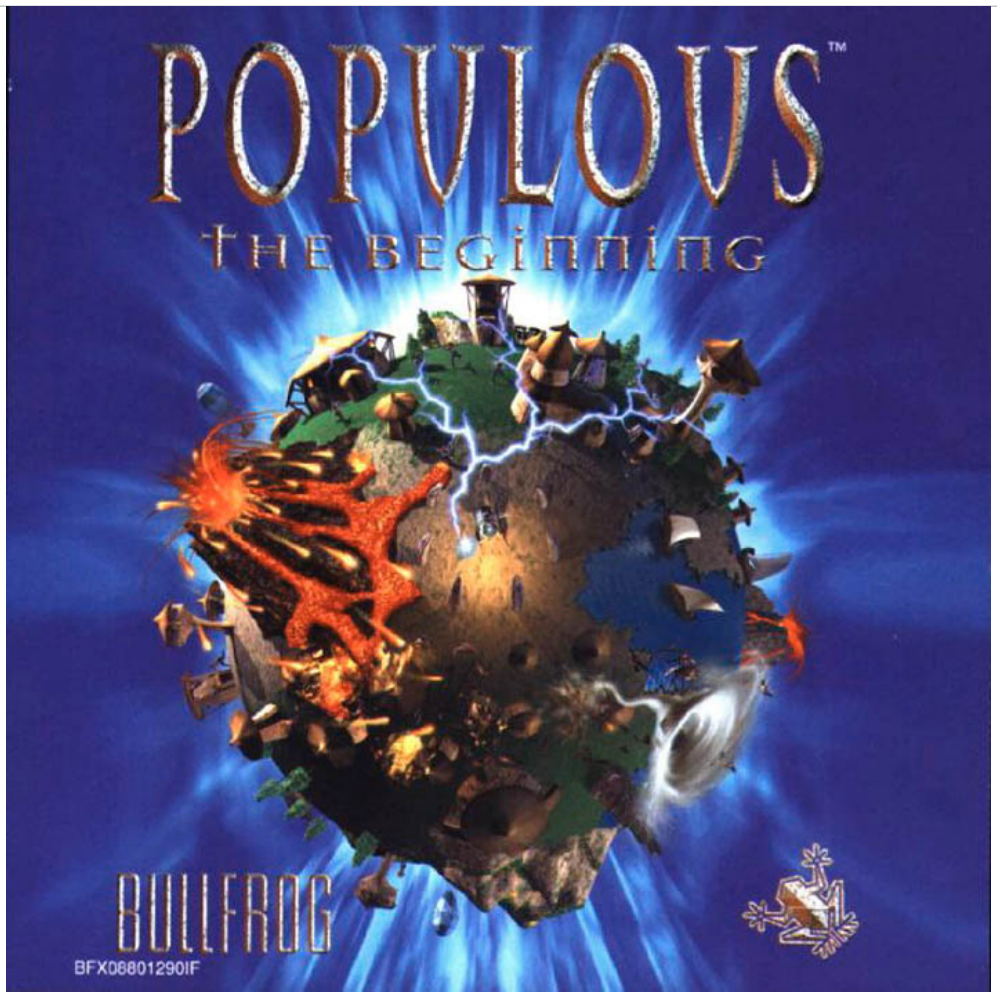
Designer example - Peter Molyneux

- well known example of a designer who pushed boundaries
  - *in particular, what we perceive as a game*
- breakthrough moment came with the design of the game **Populous**
  - *effectively created the **god** gaming genre*
- **Populous** was released in 1989 by his company **Bullfrog**
  - *sold over 4 million copies*
  - *best version originally released on the Commodore Amiga*
- **Black and White** game for Windows PCs released in 2001
  - *known for its unique design and gameplay*
  - *its overall depth and scope*
  - *renowned for its creatures' artificial intelligence*
  - *set a new Guinness World Record for its overall complexity*
- he also created game series such as
  - *Dungeon Keeper*
  - *Theme Park*
  - *Fable*
  - *The Trail*
  - ...

Image - Peter Molyneux

Peter Molyneux

Image - Populous - 1989

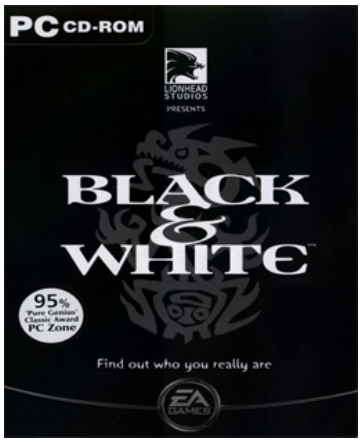


Populous cover

Video - Populous - Amiga

Source - Populous on the Amiga, Youtube

Image - Black and White - 2001

Black cover	White cover
	

Video - Peter Molyneux's Black and White

Source - Black and White review, YouTube

A bit of fun - Populous II

Trials of the Olympian Gods

- Play online - Populous II

Python and Pygame - Game Example 1

add graphics to the sprites

- now start to add some custom images for our sprite objects
  - *player object, mobs, projectiles, and a game background...*
- add images and backgrounds to our shooter game to help represent objects
  - *player's ship, laser beams firing, asteroids to hit, and star-filled background*
- before we can add our images for the sprites and backgrounds
  - *need to add some images files to our game's directory structure*
  - *normally create an `assets` folder*
  - *add any required images, audio, video &c. for our game...*
- may now update our directory structure to include the required `assets`,

```
|-- shootemup
    |-- assets
        |-- images
            |__ ship.png
```

Video - Add Graphics

add images to the game

Python and Pygame - Game Example I

import game assets

- need to import the Python module for `os`
- allows us to query a local OS's directory structure.

```
# import os
import os
```

- specify the directory location of the main game file
  - *so Python can keep track of the relative location of this file, e.g.*

```
game_dir = os.path.dirname(__file__)
```

- `__file__` is used by Python to abstract the root application file
  - *then portable from system to system*
  - *allows us to set relative paths for game directories, e.g.*

```
# game assets
game_dir = os.path.dirname(__file__)
# relative path to assets dir
assets_dir = os.path.join(game_dir, "assets")
# relative path to image dir
img_dir = os.path.join(assets_dir, "images")
```

- may then import an image for use as a sprite as follows,

```
# assets - images
ship = pygame.image.load(os.path.join(img_dir, "ship.png"))
```

convert and colour key

- as we import an image for use as a sprite within our game
  - *need to use a `convert()` method*
  - *ensures image file is of a type Pygame can use natively*
- if not, there is a potential for the game to perform more slowly
- convert example,

```
ship = pygame.image.load(os.path.join(img_dir, "ship.png")).convert()
```

- for each image that Pygame adds as a sprite
  - *a bounding rectangle will be set with a given colour*
- in most examples, we want to set the background of our sprite to transparent
- rectangle for the image will now blend with the background colour of our game window, e.g.

```
ball.set_colorkey(WHITE)
```

- now check for white coloured pixels in the image background
  - *then set them to transparent*

Python and Pygame - Game Example 1

add game background

- now add a background image for our game
  - *we might recreate stars and space for our game window, e.g.*

```
# load graphics
bg_img = pygame.image.load(os.path.join(img_dir, "bg-purple.png")).convert
```

- also add a rectangle to contain our background image

```
# add rect for bg - helps locate background
bg_rect = bg_img.get_rect()
```

- basically helps us know where to add our background image
  - *then subsequently find it as needed with the logic of our game*
  - *then draw our background image as part of the game loop, e.g.*

```
# draw background image - specify image file and rect to load image
window.blit(bg_img, bg_rect)
```

Python and Pygame - Game Example 1

add game images

- need to add an image for our player's ship, laser beams, and asteroids to shoot, e.g.

```
# add ship image
ship_img = pygame.image.load(os.path.join(img_dir, "ship-blue.png")).convert()
# ship's laser
laser_img = pygame.image.load(os.path.join(img_dir, "laser-blue.png")).convert()
# asteroid
asteroid_img = pygame.image.load(os.path.join(img_dir, "asteroid-med-grey.png")).convert()
```

- to use these new images in our game
  - *need to modify the code for each object, e.g. `Player` object*
  - *update our class to include a reference to the `ship_img`*

```
self.image = ship_img
```

- also customise this image by scaling it to better fit our game window, e.g.

```
# load ship image & scale to fit game window...
self.image = pygame.transform.scale(ship_img, (49, 37))
# set colorkey to remove black background for ship's rect
self.image.set_colorkey(BLACK)
```

- also update our ship's rect using a `colorkey`
  - *ensures black `rect` is not visible in the game window*

resources

- notes = graphics-and-sprites.pdf
- code = graphicssprites1.py

game example

- shooter0.4.py
  - *add graphics for sprites*
    - images for player's ship, ship's laser, and asteroids &c.
    - set `colorkey` for rect of sprite's
    - set background image for game window...

Video - Shooter0.4

add graphics for sprites

Games and Ideas

express ideas in video games - part 1

- often begin game development by representing behaviour and structure of real-world system
  - *e.g. cars driving, people walking, planes flying...*
  - *such systems are apparent throughout our games*
- begin building our game
  - *usually start with a known model of our chosen system*
  - *also coding potential outcomes*
  - *one of the inherent features of coding and development*
- such outcomes are developed to meet the defined requirements for a set of rules
  - *usually those defined for the system itself*
  - *or combined with the rules of the game*
- J. Murray, in 1997
  - *referred to this simply as a* ***procedural representation***
  - *video games are good at this type of representation*
- classic example of such procedural representation is the popular game *Sim City*
  - *models urban development, planning, general dynamics of city and urban living...*
  - *able to model societal and cultural patterns within this urban environment*
    - e.g. crime rates, pollution levels, economy...
- Ian Bogost explains that

> *"video games represent processes in the material world - war, urban planning, sports, and so forth- and create new possibility spaces for exploring those topics."*

Bogost, I, *The Rhetoric of Video Games*. in *The Ecology of Games*... Salen, E. MIT Press. Cambridge, MA. 2008.

Games and Ideas

express ideas in video games - part 2

- as we begin development of our game
  - *we are expressing ideas of a given system*
  - *often in a procedural manner*
- as our players experience the game
  - *they begin to form an impression or idea of the system itself*
  - *the underlying system being represented*
- the game has started to impart its ideas upon the player
- designers and developers represent their own interpretations and impressions
  - *of the underlying real-world system in the game*
- does this system actually exist in the first place?
  - *Bogost, I. has argued such video game systems inherently speculative*
  - *derived from the developer, not directly from the system itself*
- such subjectivity naturally creates a tension and dissonance, according to Bogost, I.
  - *between the player's pre-conceptions of a system*
  - *and the developer's implementation*
- tension helps express the game itself, encouraging a player to
  - *explore*
  - *question*
  - *and test the game's own systems, concepts, and general gameplay*
- can be a valuable reason to continue playing the game

Games and Ideas

express ideas in video games - part 3

- Bogost describes models as a good form of representing procedural game play
- Sid Meir's **Civilization** series of games
  - *each game can be thought of in terms of a model*
  - *a model of how real world, perceived global affairs occur...*
- specifics of the game may use ancient history and societies its model
  - *may serve as a model of many principles governing international relations today*
  - *game processes, logical outcomes reflect known world operations*
- each game uses a procedural model
  - *a player still maintains a certain degree of agency*
- player's gameplay procedure may affect the experience
  - *to an equal extent as the game's procedure...*
- each game provides an opportunity to interpret systems, rules, and procedures
- player may decide how to interpret and modify their meaning
  - *within their gameplay and experience...*
- *Civilization* series is a great example of **procedural representation** in gaming

Video - Procedural Representation

Civilization series

Source - Sid Meier's Civilization, Youtube

Video - Procedural Representation

Animal Crossing

Source - Animal Crossing, YouTube

Resources

Demos

- pygame graphics and sprites
- graphicssprites1.py
- pygame - Game 1 Example
- shooter0.4.py

Games

- Abzu
- Journey
- Animal Crossing
- Black and White
- Civilization series
- Populous
- Proteus

Game notes

- Pygame
- graphics-and-sprites.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.
- Pygame
- pygame.event
- pygame.key
- pygame.locals
- various
- God Game
- Peter Molyneux
- Populous

Videos

- Abzu trailer - YouTube
- Animal Crossing
- Black and White review - YouTube
- Populous on the Amiga - Youtube
- Sid Meier's Civilization, Youtube