Two Generic Princesses (2GP)

Make More Engines, Inc.
Week 6

Review: Game in a Nutshell

Two generic princesses, a warrior and a witch, have lost their friend and must navigate a dangerous dungeon to save her.

Puzzling puzzles and evil enemies stand in the way of our hapless heroines, but armed with a sparkly wand and brute force, the princesses will prevail… with flair.

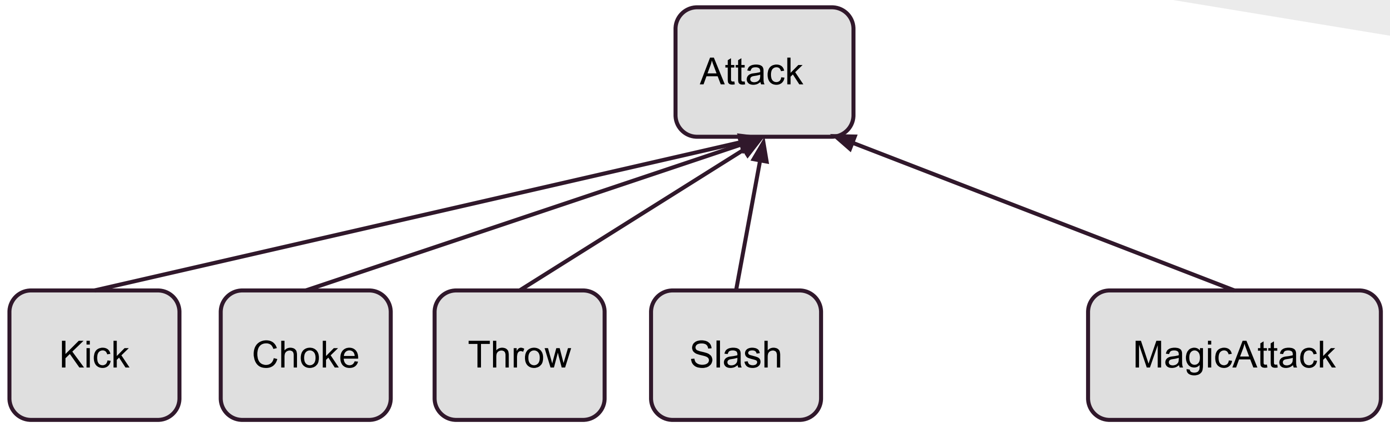
Review: Control Scheme

		Warrior	**Witch**
	Kinect	Special	--
	MIDI	--	Special
X Bo x	**Shoulder**	Block	cancel spell
	Trigger	Basic attack	
	C-Stick	Move	
	A B X Y	Special (temporary, for testing)	

Sprites and Animations

- animated special attacks for warrior princess (kick, slash, throw, roll)
- need special mechanism for choke hold - attach enemy entity to princess
- Attack class has invisible projectiles (no images assigned yet)

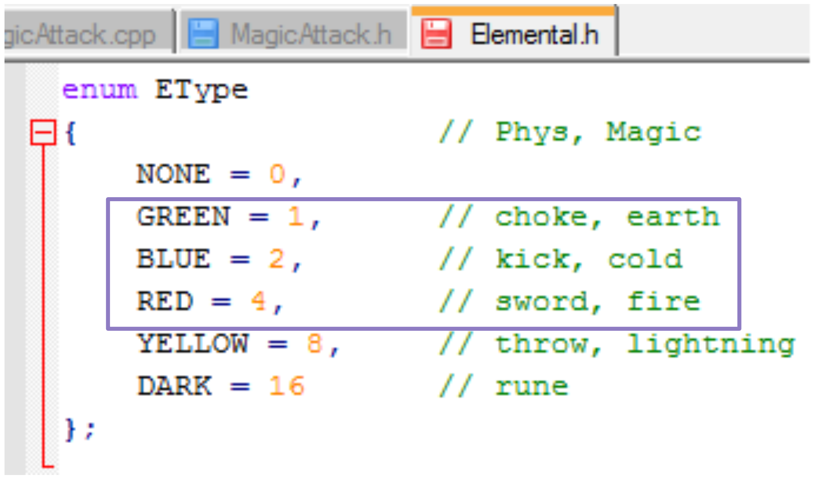
Attack Class: Subclassing



Physical attacks can cause one of three effects

Magical attacks can cause a mix of all three, and is a more generic class than the individual physical attacks

Attack Class: Status Effects



```
enum EType
{                       // Phys, Magic
    NONE = 0,
    GREEN = 1,          // choke, earth
    BLUE = 2,           // kick, cold
    RED = 4,            // sword, fire
    YELLOW = 8,         // throw, lightning
    DARK = 16           // rune
};
```

Attack Class: Creativity Boost

Physical: Left and right poses work the same way; award bonus damage for changing it up between attacks

Musical: All Gb notes will queue Earth, Bb queues fire, etc.; award bonus damage for using a different Gb or Bb between attacks

Enemy Classes

Immunity

Tint green, blue, or red

Red enemies ignore all fire and sword damage

AI

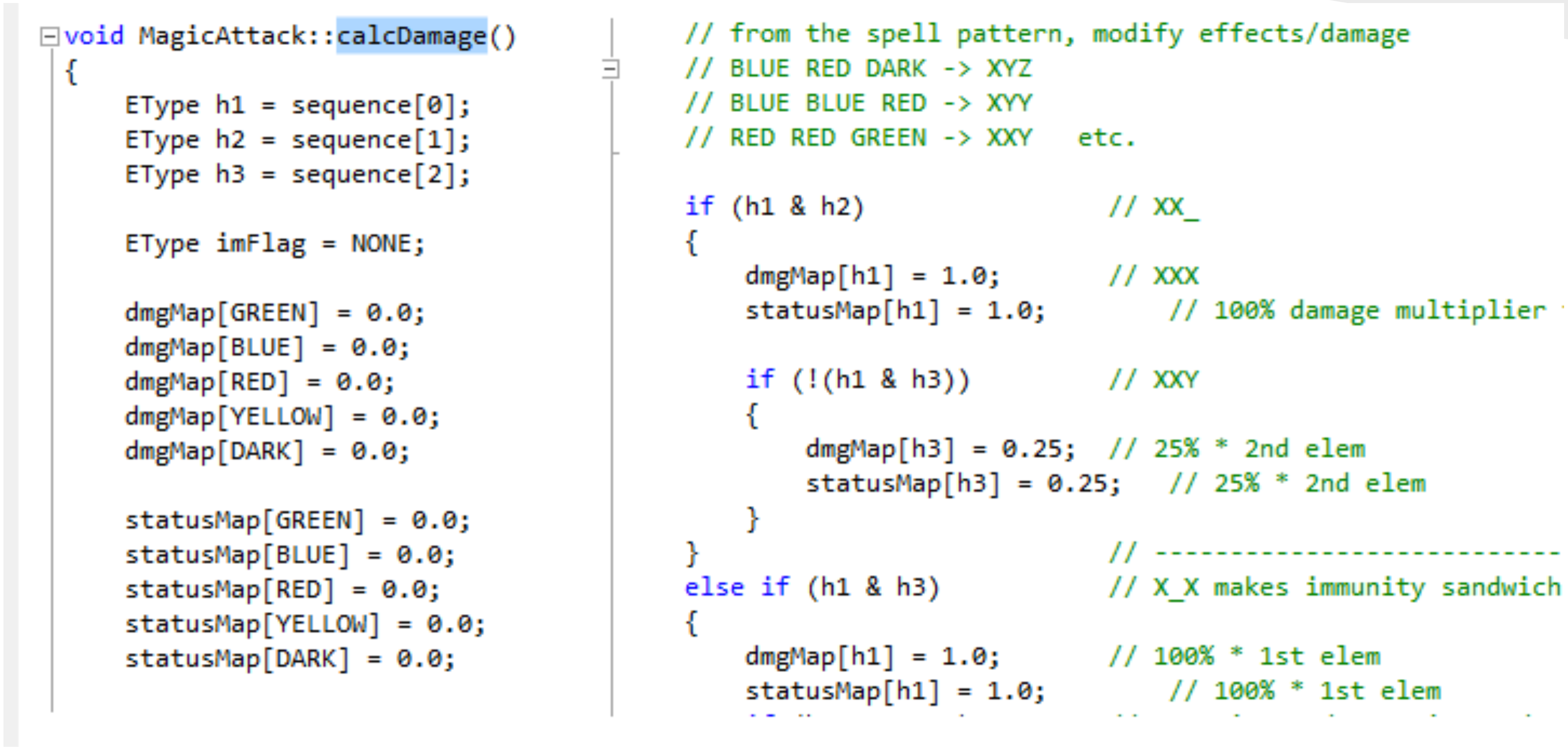
Near, far,
wherever you are
I believe that
the loop does go on

just kidding

Magic Attacks: Elements

Earth – calls up the vines from the ground to bind your enemies in place.
Cold – chills enemies to slow their movement speed.
Fire – burns enemies, dealing damage over time.
Lightning – arcs between enemies standing close to each other.
Dark – leaves a rune which explodes when an enemy steps on it.

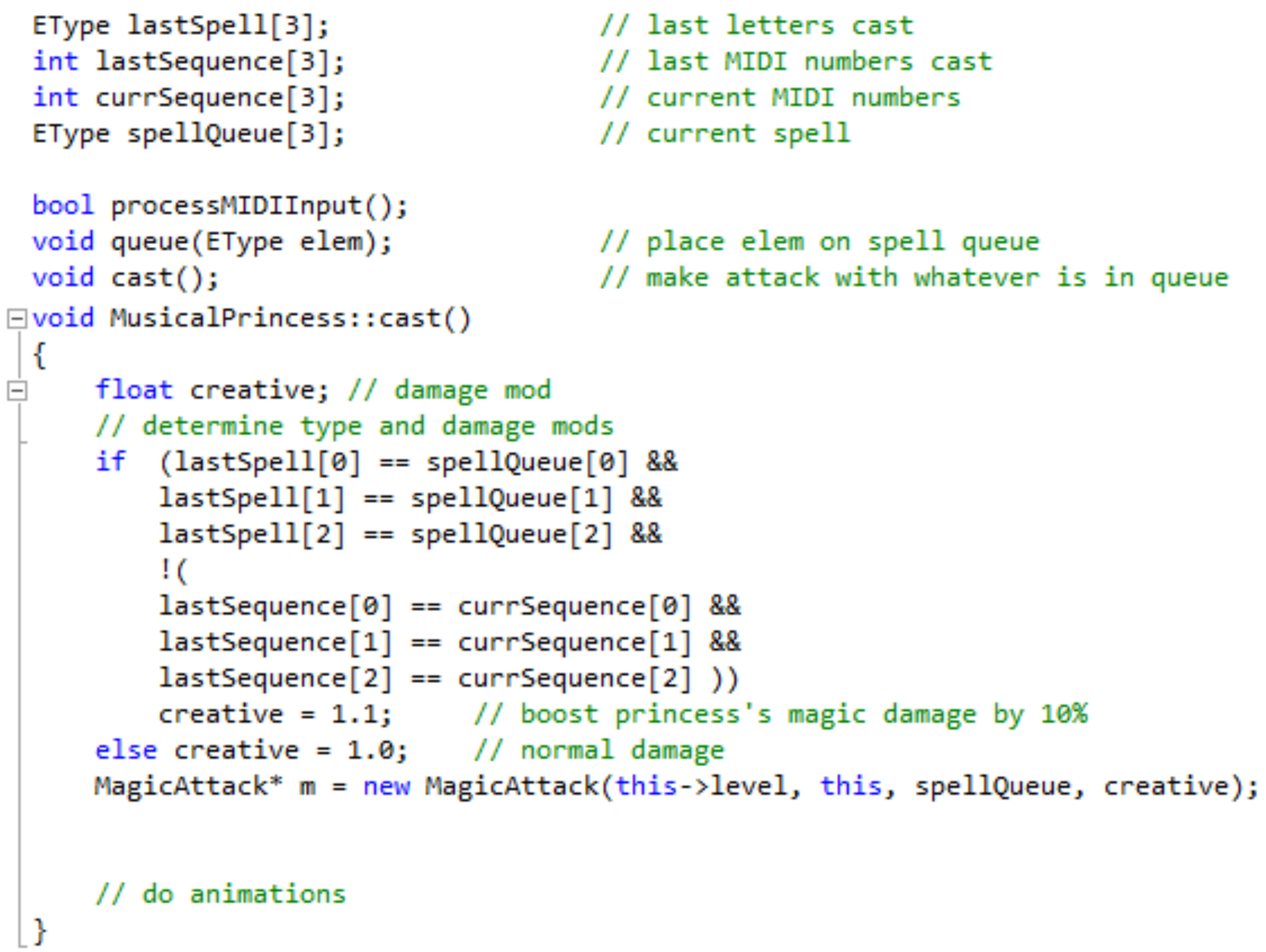
Magic Attacks: Combinations



```
void MagicAttack::calcDamage()
{
    EType h1 = sequence[0];
    EType h2 = sequence[1];
    EType h3 = sequence[2];

    EType imFlag = NONE;

    dmgMap[GREEN] = 0.0;
    dmgMap[BLUE] = 0.0;
    dmgMap[RED] = 0.0;
    dmgMap[YELLOW] = 0.0;
    dmgMap[DARK] = 0.0;

    statusMap[GREEN] = 0.0;
    statusMap[BLUE] = 0.0;
    statusMap[RED] = 0.0;
    statusMap[YELLOW] = 0.0;
    statusMap[DARK] = 0.0;
```

```
    // from the spell pattern, modify effects/damage
    // BLUE RED DARK -> XYZ
    // BLUE BLUE RED -> XYY
    // RED RED GREEN -> XXY   etc.

    if (h1 & h2)                // XX_
    {
        dmgMap[h1] = 1.0;       // XXX
        statusMap[h1] = 1.0;        // 100% damage multiplier

        if (!(h1 & h3))         // XXY
        {
            dmgMap[h3] = 0.25;  // 25% * 2nd elem
            statusMap[h3] = 0.25;   // 25% * 2nd elem
        }
    }                           // ----------------------------
    else if (h1 & h3)           // X_X makes immunity sandwich
    {
        dmgMap[h1] = 1.0;       // 100% * 1st elem
        statusMap[h1] = 1.0;        // 100% * 1st elem
```

Magic Attacks: Execution

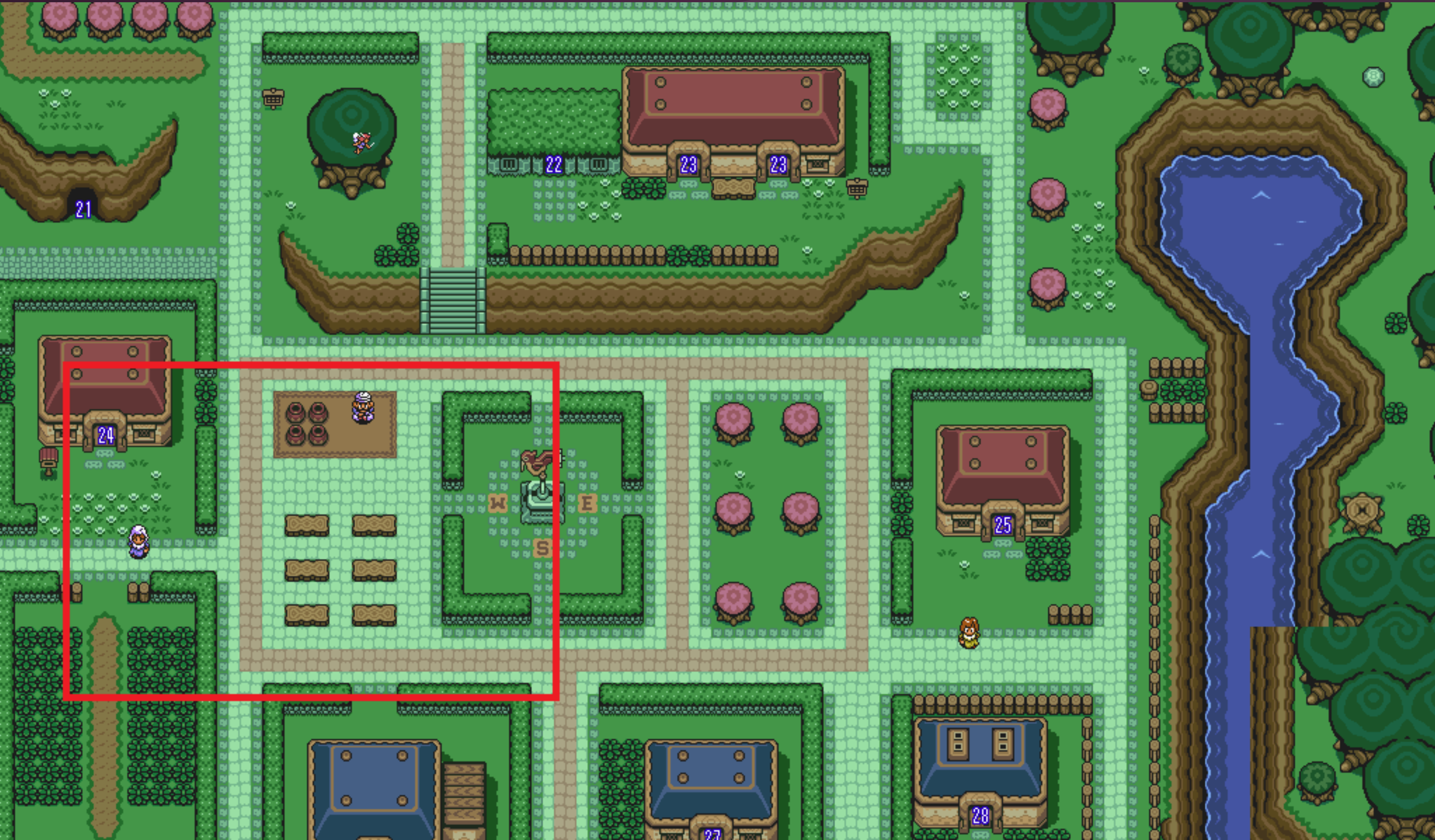


```
EType lastSpell[3];                     // last letters cast
int lastSequence[3];                    // last MIDI numbers cast
int currSequence[3];                    // current MIDI numbers
EType spellQueue[3];                    // current spell

bool processMIDIInput();
void queue(EType elem);                 // place elem on spell queue
void cast();                            // make attack with whatever is in queue
void MusicalPrincess::cast()
{
    float creative; // damage mod
    // determine type and damage mods
    if  (lastSpell[0] == spellQueue[0] &&
        lastSpell[1] == spellQueue[1] &&
        lastSpell[2] == spellQueue[2] &&
        !(
        lastSequence[0] == currSequence[0] &&
        lastSequence[1] == currSequence[1] &&
        lastSequence[2] == currSequence[2] ))
        creative = 1.1;     // boost princess's magic damage by 10%
    else creative = 1.0;    // normal damage
    MagicAttack* m = new MagicAttack(this->level, this, spellQueue, creative);


    // do animations
}
```

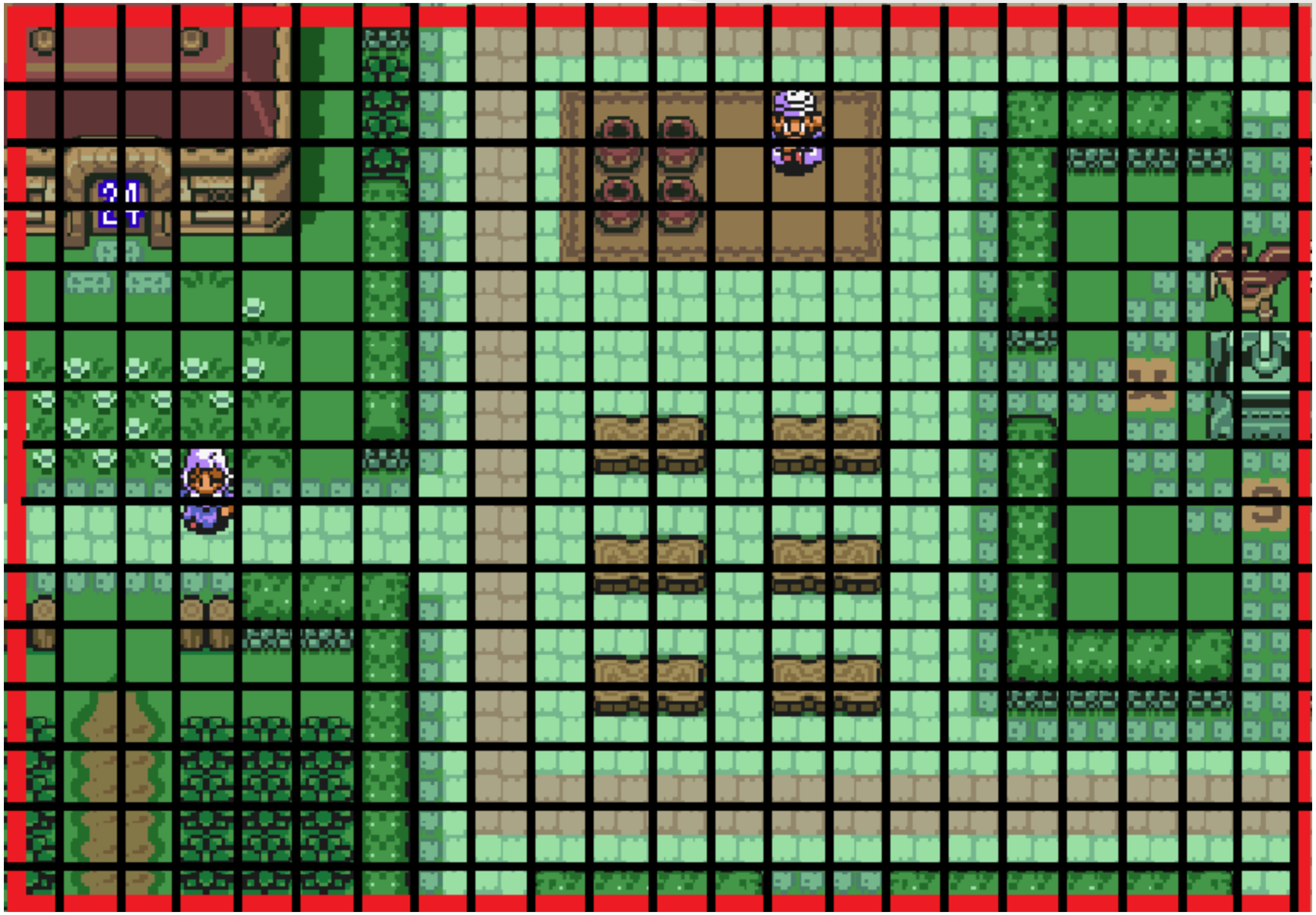
Physical Attacks

Choke – Enemies are dragged around .
Kick – Enemies are slowed.
Slash – Enemies lose health over time.
Throw – Long-range attack.
Roll – Move quickly to catch up to or avoid an enemy.

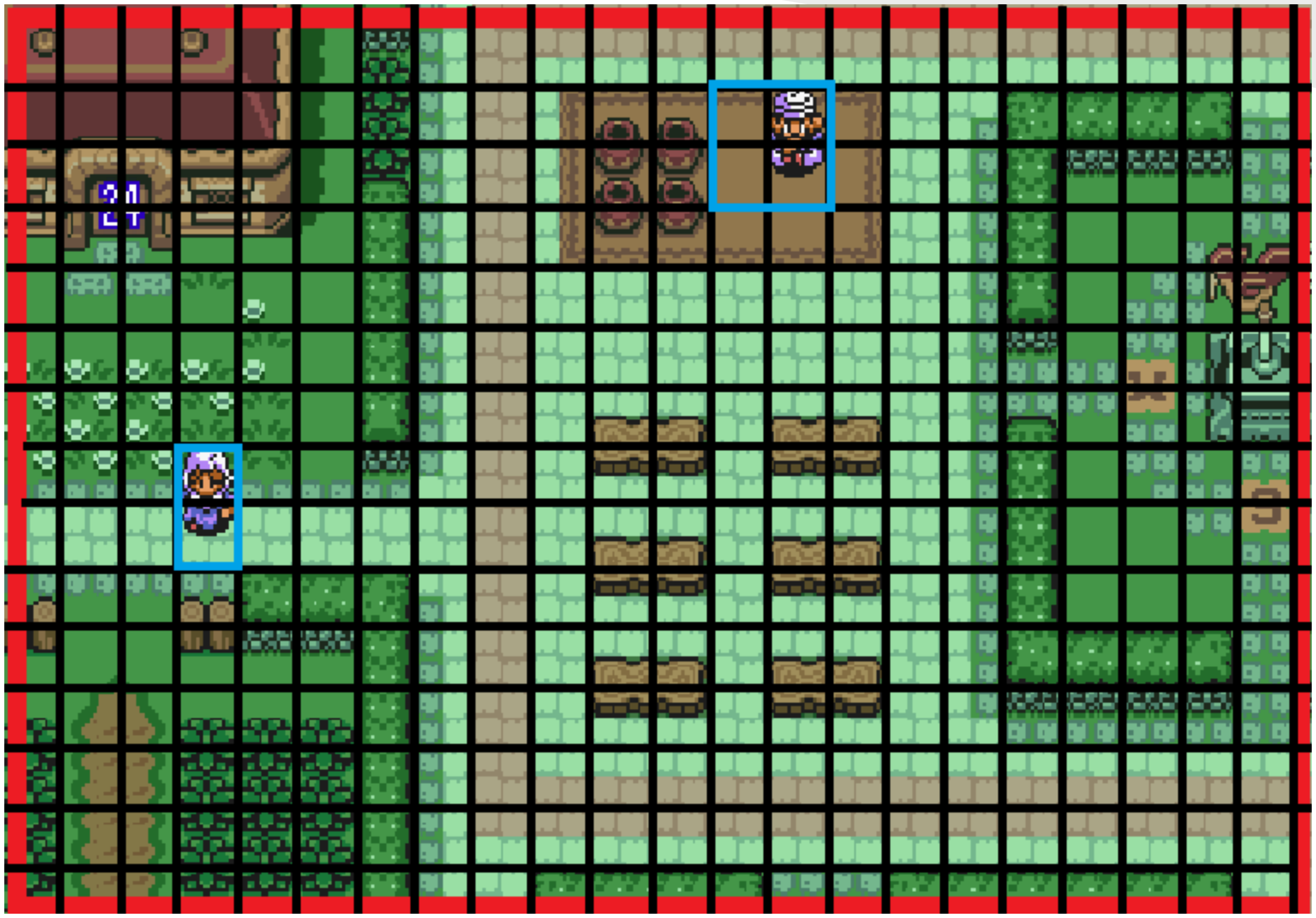
Camera



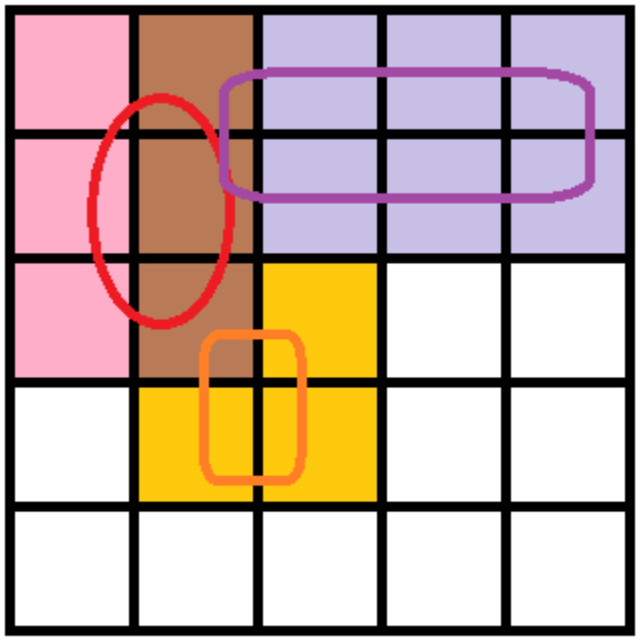
Closeup of tiles



Hashing objects into tiles



Hashing objects into tiles



🟫 Multiple Objects in one Tile

Rendering

-Draw each tile in the camera view
-For each layer of rendering…
    -For each tile in the view, add occupying entities to a set
-Sort objects in set by Y value
-draw objects

Collisions

-For each entity within a boundary that slightly exceeds camera bounds…

- -Check each tile the entity occupies for any other entities
- -For each other entity found…
  - -Do accurate collision detection between objects
  - -Resolve collisions

Attacks

-Subclass of entity
-When collision is detected
    -collide with princess
    -collide with enemy
    -collide with attack