# B2 Level Design -Mechanics and Layout

Once again biology is proving a lot harder to include than physics was, as such, B2 currently has the most interactive puzzles, each one having multiple different stages for the puzzle to be complete.

The B2 level is based on Topic 3: The building blocks of life:

-The magic microscope (Inside a bacteria cell).

-Acid match (Protein manufacture).

-The enzyme gate (Enzyme action).

## **Name**: B2

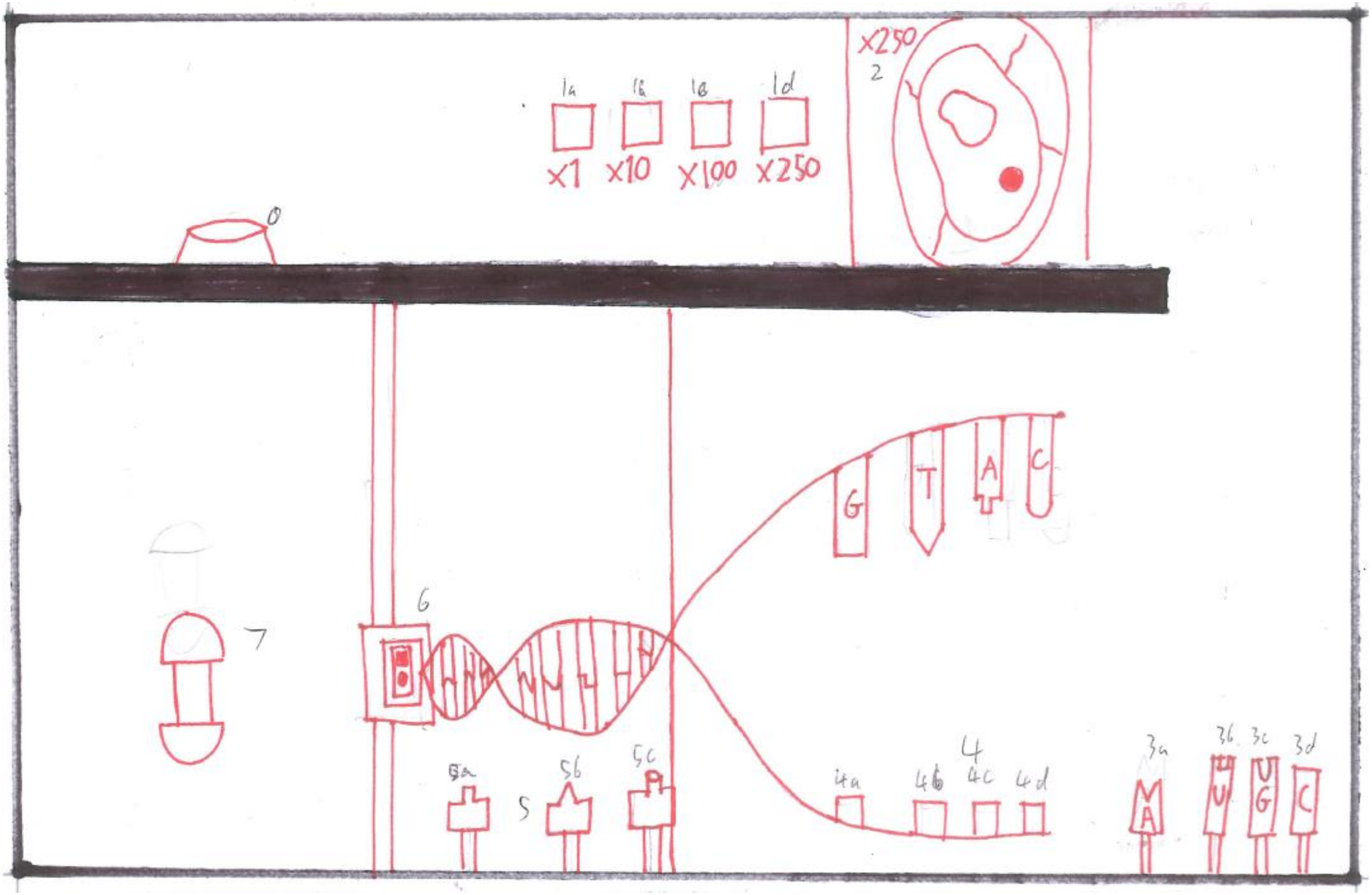
**Class**:  Stage
**States**: Incomplete, Complete
**Can transfer states**: no
**Starting state**: Incomplete
**Algorithm(s):**

```
If [B.Teleporter]: {on},
And [B2]: not {Incomplete},
enter {Incomplete}.
While: {Incomplete},
[B2] will be assigned to main instance.
If [Battery]: {Win},
enter {complete}.
While: {complete},
[B2] will be deleted.
```

## **Name**: B2.Teleporter

**Class**:  Stage object
**States**: Off, on
**Can transfer states**: no
**Starting state**: Off
**Algorithm(s):**

```
animation: teleporter
If [B2]: {Off};
Load [Player] in +0 spaces
```


x250
2
1a
1b
1c
1d
x1
x10
x100
x250
0
G
T
A
C
6
7
5a
5b
5c
5
4
4a
4b
4c
4d
3a
3b
3c
3d
A
U
G
C

1. *Buttons*: Object 1 consists of four buttons, each on labeled with their respective zoom. When the player interacts with the button, object 1 will change state depending on the button (states: 1, 10, 100 and 250).

## **Name**: Buttons

**Class**:  Interactive Object
**States**: 1, 10, 100, 250
**Can transfer states**: no
**Starting state**: 250
**Algorithm(s):**
While: {1}; animation: 1.
**1b**
If {interacting} occurs within +-1 spaces,
enter {10}.
While: {10}; animation: 10.
**1c**
If {interacting} occurs within +-1 spaces,
enter {100}.
While: {100}; animation: 100.
**1d**
If {interacting} occurs within +-1 spaces,
enter {250}.
While: {250}; animation: 250.
**1a**
If {interacting} occurs within +-1 spaces,
enter {1}.
While: {1}; animation: 1.

2. *Microscope*: Starts in an off state. Depending on the state of object 1, object 2 will change its animation, when object 1 is in a 1 state, object 2 will enter an empty state, changing its animation and removing all interactions.

**Name**: Microscope
**Class**: Reactive Object
**States**: 1, 10, 100, 250
**Can transfer states**: no
**Starting state**: 1
**Algorithm(s):**

```
If [Buttons]: {250}
enter {250}.
While: {250}; animation: 250.
Mimic {[platform]}.
If [Buttons]: {10}
enter {10}.
While: {10}; animation: 10.
Mimic {[platform]}.
If [Buttons]: {100}
enter {100}.
While: {100}; animation: 100.
Mimic {[platform]}.
If [Buttons]: {1}
enter {1}.
While: {1}; animation: 1.
Mimic {[none]}.
```

3. *Acids*: Object 3 consists of four amino acids, the player can interact with any of the four to enter a hold state (states: A, U, G and C). The player can transfer any of the four states back by interacting with object 3.

## **Name**: Acids

**Class**: Hold Object
**States**: Idle, A, U, G, C
**Can transfer states**: Yes (can hold multiple states)
**Starting state**: Idle
**Algorithm(s):**

```
While: {Idle}; animation: Idle.
3a
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {A} to [Player].
While: {A}; animation: A.
While: {A},
If {interacting} occurs within +-1 spaces,
Transfer {A} from [Player].
Enter {Idle}
While: {Idle}; animation: Idle.
3b
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {U} to [Player].
While: {U}; animation: U.
While: {U},
If {interacting} occurs within +-1 spaces,
Transfer {U} from [Player].
Enter {Idle}
While: {Idle}; animation: Idle.
3c
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {G} to [Player].
While: {G}; animation: G.
While: {G},
If {interacting} occurs within +-1 spaces,
Transfer {G} from [Player].
Enter {Idle}
While: {Idle}; animation: Idle.
3d
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {C} to [Player].
While: {C}; animation: C.
While: {C},
If {interacting} occurs within +-1 spaces,
Transfer {C} from [Player].
Enter {Idle}
While: {Idle}; animation: Idle.
```

4. *RNA*: Starts in an off state, the player can transfer states into the four parts of object 4. When 4a is in a C state, 4b is in an A state, 4c is in a U state and 4d is in a G state, then object 4 will enter an on state. While in this state, object 4 changes it's interactions and its animation to allow the player to pass.

## **Name**: RNA

**Class**: Interactive Object
**States**: Idle, A, U, G, C, On, Off
**Can transfer states**: Yes
**Starting state**: Off
**Algorithm(s):**
While: {Off}; animation: Off,
Mimic {[platform]}.
If [4a, 4b, 4c, 4d]: {On},
enter {On}.
While: {On}; animation: On.
Mimic {[none]}.
**4a**
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {C} from[Player].
While: {C}; animation: C.
While: {C},
If {interacting} occurs within +-1 spaces,
Transfer {C} to [Player].
Enter {On}
While: {On}; animation: On.
**4b**
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {A} to [Player].
While: {A}; animation: A.
While: {A},
If {interacting} occurs within +-1 spaces,
Transfer {A} from [Player].
Enter {Idle}
While: {On}; animation: On.
**4c**
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {U} from [Player].
While: {U}; animation: U.
While: {U},
If {interacting} occurs within +-1 spaces,
Transfer {U} to [Player].
Enter {On}
While: {On}; animation: On.
**4d**
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {G} from [Player].

While: {G}; animation: G.
While: {G},
If {interacting} occurs within +-1 spaces,
Transfer {G} to [Player].
Enter {On}
While: {On}; animation: On.

5. *Proteins*: Object 5 consists of three proteins, the player can interact with any of the 3 to enter a hold state (states: red, blue and green (they will be coloured as such)). The player can transfer any of the three states back by interacting with object 5.

## **Name**: Proteins

**Class**: Hold Object
**States**: Idle, red, blue, green
**Can transfer states**: Yes (can hold multiple states)
**Starting state**: Idle
**Algorithm(s):**
While: {Idle}; animation: Idle.
**5a**
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer red} to [Player].
While: {red}; animation: red.
While: {red},
If {interacting} occurs within +-1 spaces,
Transfer {red} from [Player].
Enter {Idle}
While: {Idle}; animation: Idle.
**5b**
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {blue} to [Player].
While: {blue}; animation: blue.
While: {blue},
If {interacting} occurs within +-1 spaces,
Transfer {blue} from [Player].
Enter {Idle}
While: {Idle}; animation: Idle.
**5c**
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {green} to [Player].
While: {green}; animation: green.
While: {green},
If {interacting} occurs within +-1 spaces,
Transfer {green} from [Player].
Enter {Idle}
While: {Idle}; animation: Idle.

6. *Enzyme gate*: Starts in a locked state. the player can transfer states onto object 6. When in a red or green state object 6 will change animation. If the user tries to transfer the blue state, they player will say (It doesn't fit). When object 6 enters and red and green state, it will change to an unlocked state. While in this state, object 6 changes it's interactions and its animation to allow the player to pass.

**Name**: Enzyme gate
**Class**: Interactive Object
**States**: Idle, On, Off, red, blue, green
**Can transfer states**: Yes
**Starting state**: Off
**Algorithm(s):**

```
While: {Off}; animation: Off,
Mimic {[platform]}.
If [6a, 6b]: {On},
enter {On}.
While: {On}; animation: On.
Mimic {[none]}.
```

**6a**

```
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {red} from[Player].
While: {red}; animation: red.
While: {red},
If {interacting} occurs within +-1 spaces,
Transfer {red} to [Player].
Enter {On}
While: {On}; animation: On.
```

**6b**

```
While: {Idle},
If {interacting} occurs within +-1 spaces,
Transfer {green} to [Player].
While: {green}; animation: green.
While: {green},
If {interacting} occurs within +-1 spaces,
Transfer green} from [Player].
Enter {Idle}
While: {On}; animation: On.
```

**6c**

```
While: {Idle},
If {interacting} occurs within +-1 spaces,
While [Player]: {Blue},
[Player] speak (It doesn't fit).
```

7. Battery: When the player touches object 7 the level instance will enter a win state. This will end and lock the instance while opening the neutral area in a B2 clear state (This will change certain visuals of the neutral area).

## **Name**: B2 Battery

**Class**: Stage Object
**States**: On, Off, Win
**Can transfer states**: no
**Starting state**: On
**Algorithm(s):**

```
While: {On}; animation: On.
If [Player] enters +-0 spaces,
enter {Off}.
While: {Off}; animation: Off.
If {Off},
Wait 5seconds
enter {Win}.
```