README DEVELOPER — Step 2

Overview

This project is intentionally grown in **small, beginner-friendly steps**.
Step 1 gave us three files:

- `settings.py` → **all knobs in one place** (screen size, colors, speeds)
- `models.py` → **tiny typed data containers** (currently just `Laser`)
- `game.py` → **main game loop** with helper functions and comments

In **Step 2** we split responsibilities further by introducing *sprite-like* classes and a dedicated UI module. The old `game_objects.py` module is no longer used.

File Layout (Step 2)

```
.
├── settings.py         # Tunable constants (screen, rocket, lasers, UI)
├── models.py           # Lightweight dataclasses (e.g. Laser)
├── sprites.py          # Player, Enemy, Explosion classes
├── ui.py               # Score rendering, Game Over screen
├── game.py             # Main loop (now delegates more work)
└── README\_DEVELOPER.md
```

sprites.py

- `Player`
  - Owns position (x, y)
  - Handles input (move up/down)
  - Renders itself (`draw(surface)`)
  - Creates new `Laser` objects when shooting
- `Enemy`
  - Owns position and speed
  - Moves left every frame
  - Knows when it reached the screen edge (`reached_end()`)
  - Renders itself (`draw(surface)`)
- `Explosion`
  - Starts at position with radius 0
  - Grows each frame until max radius
  - Renders itself (`draw(surface)`)

ui.py

- Score drawing (top right)
- Game Over screen (blocking until Enter)
- Any future HUD elements (lives, high score, etc.)

game.py

Now mostly orchestration:

1. Setup (screen, stars, fonts)
2. Create initial `Player` and `Enemy`
3. Loop:
   - Handle input
   - Update all sprites
   - Detect collisions
   - Draw background, sprites, UI
   - Manage round/game over
4. Restart flow (round → game over → round)

Coding Conventions

- **Type hints everywhere** (ints, tuples, lists).
- **Docstrings** in Google style:

  ```
  def move(self, dy: int) -> None:
      """Move the player vertically.

      Args:
          dy (int): Number of pixels to move. Positive = down, negative = up.
      """
  ```

```


* **Constants** only in `settings.py` — no hard-coded numbers in logic.
* **Drawing code** lives with the object (e.g., `Player.draw()`).
* **Updates** are small, explicit steps (avoid hidden side-effects).
* **One responsibility per module.**

---

## Next Steps After Step 2

* Step 3 — Add sound effects (shoot, explosion, game over).
* Step 4 — Add menu screen and multiple lives.
* Step 5 — Add difficulty modes or level progression.

---
```

~~~~
## Tips for Contributors

* If you’re new: start by tweaking numbers in `settings.py`.
* If you’re adding features: decide whether it’s **game logic** (`game.py`), **an
object** (`sprites.py`), or **UI** (`ui.py`).
* Always run the game after a change:

  ```bash
  python game.py
  ```
* Keep commits small — one feature or one refactor at a time.
* Don’t use `game_objects.py` anymore — it has been fully replaced.

```
~~~~