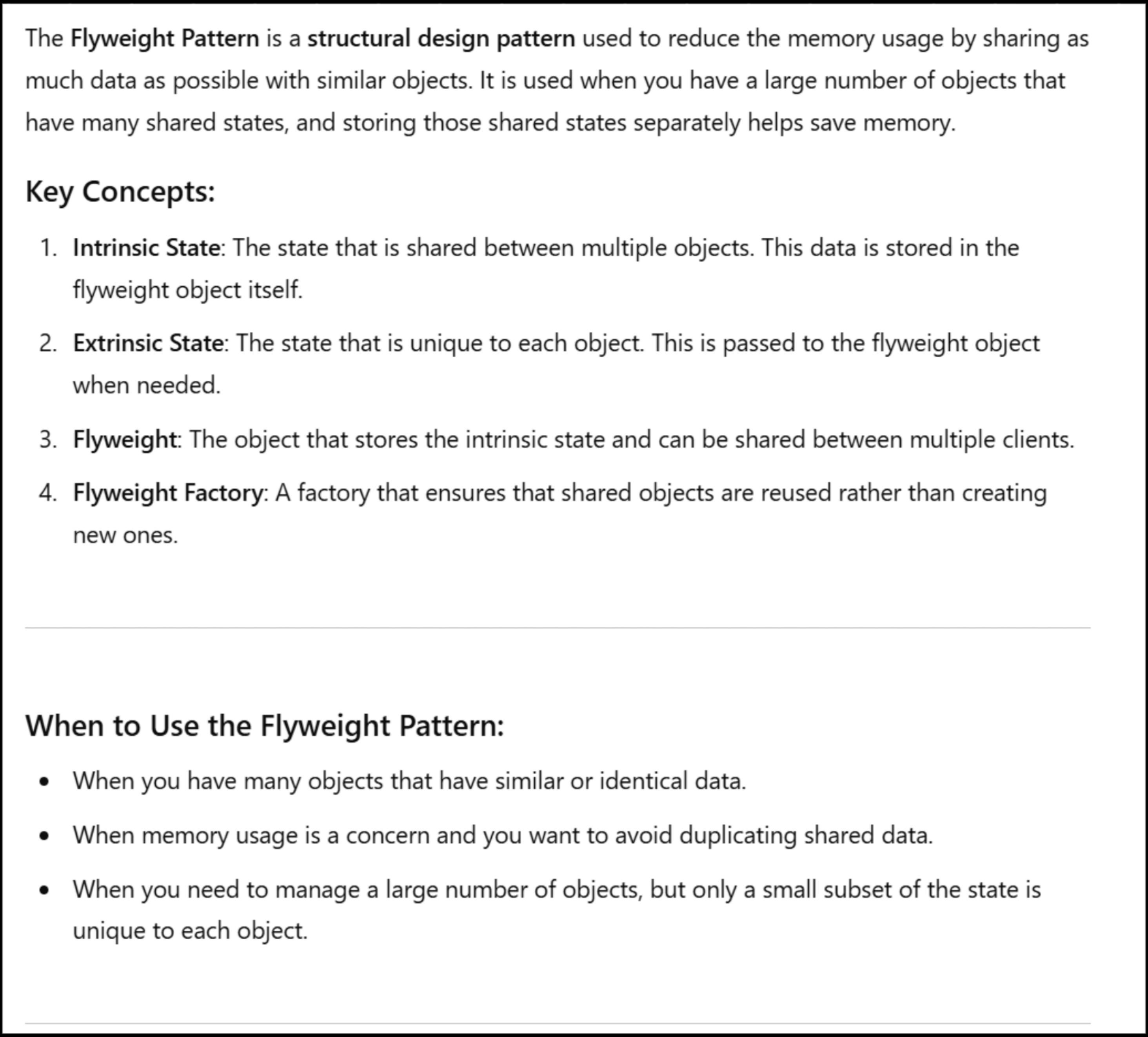
[Flyweight Design Pattern]



The **Flyweight Pattern** is a **structural design pattern** used to reduce the memory usage by sharing as much data as possible with similar objects. It is used when you have a large number of objects that have many shared states, and storing those shared states separately helps save memory.

Key Concepts:

1. **Intrinsic State**: The state that is shared between multiple objects. This data is stored in the flyweight object itself.
2. **Extrinsic State**: The state that is unique to each object. This is passed to the flyweight object when needed.
3. **Flyweight**: The object that stores the intrinsic state and can be shared between multiple clients.
4. **Flyweight Factory**: A factory that ensures that shared objects are reused rather than creating new ones.

When to Use the Flyweight Pattern:

- When you have many objects that have similar or identical data.
- When memory usage is a concern and you want to avoid duplicating shared data.
- When you need to manage a large number of objects, but only a small subset of the state is unique to each object.

---

* Memozation is used here