there are no Dumb Questions

Q: How does the Hollywood Principle relate to the Dependency Inversion Principle that we learned a few chapters back?

A: The Dependency Inversion Principle teaches us to avoid the use of concrete classes and instead work as much as possible with abstractions. The Hollywood Principle is a technique for building frameworks or components so that lower-level components can be hooked into the computation, but without creating dependencies between the lower-level components and the higher-level layers. So, they both have the goal of decoupling, but the Dependency Inversion Principle makes a much stronger and more general statement about how to avoid dependencies in design.

The Hollywood Principle gives us a technique for creating designs that allow low-level structures to interoperate while preventing other classes from becoming too dependent on them.

Q: Is a low-level component disallowed from calling a method in a higher-level component?

A: Not really. In fact, a low-level component will often end up calling a method defined above it in the inheritance hierarchy purely through inheritance. But we want to avoid creating explicit circular dependencies between the low-level component and the high-level ones.

WHO DOES WHAT?

Match each pattern with its description:

Pattern	Description
Template Method	Encapsulate interchangeable behaviors and use delegation to decide which behavior to use.
Strategy	Subclasses decide how to implement steps in an algorithm.
Factory Method	Subclasses decide which concrete classes to instantiate.