Difference between Decorator and Proxy

Although decorators can have similar implementations as proxies, decorators have a different purpose. A decorator adds one or more responsibilities to an object, whereas a proxy controls access to an object.

Proxies vary in the degree to which they are implemented like a decorator. A protection proxy might be implemented exactly like a decorator. On the other hand, a remote proxy will not contain a direct reference to its real subject but only an indirect reference, such as "host ID and local address on host." A virtual proxy will start off with an indirect reference such as a file name but will eventually obtain and use a direct reference.

Popular answers indicate that a Proxy knows the concrete type of its delegate. From this quote we can see that is not always true.

The difference between Proxy and Decorator according to the GoF is that Proxy *restricts* the client. Decorator does not. Proxy may restrict what a client *does* by controlling access to functionality; or it may restrict what a client *knows* by performing actions that are invisible and unknown to the client. Decorator does the opposite: it enhances what its delegate does in a way that is visible to clients.

We might say that Proxy is a black box while Decorator is a white box.

The composition relationship between wrapper and delegate is the wrong relationship to focus on when contrasting Proxy with Decorator, because composition is the feature these two patterns have in common. The relationship between wrapper and client is what differentiates these two patterns.

- **Decorator informs and empowers its client.**
- **Proxy restricts and disempowers its client.**

Decorator Pattern focuses on dynamically adding functions to an object, while **Proxy** Pattern focuses on controlling access to an object.

EDIT:-

Relationship between a *Proxy* and the real subject is typically set at compile time, *Proxy* instantiates it in some way, whereas *Decorator* is assigned to the subject at runtime, knowing only subject's interface.