Decouple deployments from releases refers to separating the act of deploying code into production from the moment it becomes visible or accessible to users. Traditionally, deployments and releases happened simultaneously. Once the new code was deployed, it was live. However, modern DevOps practices advocate for decoupling these two processes to enhance agility, reduce risk, and improve overall software delivery.

Yes, it is absolutely possible and increasingly common to decouple deployments from releases. Techniques such as feature flags, blue-green deployments, and canary releases make this feasible [2]. For example, a new feature can be deployed to production but hidden behind a feature flag. Only when the team is ready after testing or validation is the flag toggled to make the feature live. This way, teams can push code more frequently, test in production-like environments, and minimize the risk of failures affecting all users at once.

There are few reasons not to decouple deployments and releases. In some small teams or legacy systems, the added complexity of managing feature flags [3] or infrastructure for staged rollouts might outweigh the benefits. There may also be governance or regulatory constraints that require tighter control. However, for most modern software teams, the advantages of reduced downtime, faster iteration, and safer experimentation far outweigh the drawbacks. Ultimately, decoupling gives teams more control, enabling continuous delivery while still managing business risk.

Sources:

1. <https://launchdarkly.com/blog/why-decouple-deployments-from-releases/#:~:text=Decoupling%20deployments%20from%20releases%20refers,production%20at%20the%20same%20time>.
2. <https://www.wissen.com/blog/the-role-of-blue-green-canary-and-feature-flags#:~:text=Blue%2DGreen%2C%20Canary%2C%20and%20Feature%20Flags%20are%20best%20practices,them%20close%20to%20zero%20downtime>.
3. <https://medium.com/@ss-tech/why-you-should-stop-using-feature-flags-10c3806c0b42>