2. Source control strategy

└ 2.3 Package Management

└ 2.3.1 Tools: Github Packages, Azure Artifacts

---

1. What are GitHub Packages and Azure Artifacts, and how do they differ?
2. How do you publish a package to GitHub Packages?
3. How do you publish a package to Azure Artifacts?
4. How do you authenticate to GitHub Packages and Azure Artifacts?
5. What package types are supported by GitHub Packages and Azure Artifacts?
6. How do you configure a package feed in Azure Artifacts?
7. How do you use a package from GitHub Packages in a CI/CD pipeline?
8. How do you control access and permissions for packages in GitHub Packages and Azure Artifacts?
9. How do you manage package retention and cleanup in Azure Artifacts?
10. How do you configure upstream sources in Azure Artifacts

---

1. **What are GitHub Packages and Azure Artifacts, and how do they differ?**
   - *GitHub Packages* is a package hosting service fully integrated with *GitHub* repositories, supporting multiple package types and commonly used for GitHub-based projects.
   - *Azure Artifacts* is part of Azure DevOps, offering integrated package feeds for teams, granular permissions, and support for upstream sources.
   - Key difference: Azure Artifacts is tightly integrated with Azure DevOps pipelines and boards, while GitHub Packages is GitHub-centric.

---

2. **How do you publish a package to GitHub Packages?**
   - Authenticate with a personal access token or *GitHub Actions*,
   - configure your package manager with your *GitHub* repository URL,
   - and use the package manager's publish command (e.g., `npm publish`, `dotnet nuget push`, `mvn deploy`).

---

3. **How do you publish a package to Azure Artifacts?**
   - Authenticate with *Azure Artifacts* using a personal access token or *Azure CLI*,
   - add the *Azure Artifacts* feed as a source to your package manager,
   - and use the relevant publish command (e.g., `dotnet nuget push`, `npm publish`, `twine upload` for Python).

---

4. **How do you authenticate to GitHub Packages and Azure Artifacts?**
   - For *GitHub Packages*, use a personal access token with `write:packages` scope or *GitHub Actions'* built-in `GITHUB_TOKEN`.
   - For *Azure Artifacts*, authenticate with a personal access token or Azure CLI (az artifacts universal publish/login), or with build pipeline credentials.

---

5. **What package types are supported by GitHub Packages and Azure Artifacts?**
   - *GitHub Packages* supports npm, Maven, NuGet, Docker, RubyGems, and Apache Ivy.
   - *Azure Artifacts* supports npm, NuGet, Maven, Python, and Universal Packages.

---

6. **How do you configure a package feed in Azure Artifacts?**

In *Azure DevOps*, go to `Artifacts > New Feed`, name the feed, set visibility (organization/private), optionally add upstream sources, and save. Configure your package manager (e.g., `nuget.config`, `.npmrc`) to use the feed's URL.

7. **How do you use a package from GitHub Packages in a CI/CD pipeline?**
    1. Add authentication steps to the pipeline (set up GITHUB_TOKEN or PAT)
    2. Configure the package manager with the *GitHub Packages* registry URL
    3. Install the package as usual in your pipeline steps

---

8. **How do you control access and permissions for packages in GitHub Packages and Azure Artifacts?**
    - In *GitHub Packages*, permissions are based on repository access and repository roles.
    - In *Azure Artifacts*, use feed-level permissions to grant/restrict to users, groups, or pipelines.

---

9. **How do you manage package retention and cleanup in Azure Artifacts?**

Configure retention policies in *Azure Artifacts* to automatically delete older or unlisted package versions. Go to *Artifacts > Feed > Settings > Retention policies*, and define rules for how many versions to keep and for how long.

---

10. **How do you configure upstream sources in Azure Artifacts?**

In the feed settings, add upstream sources to connect your *Azure Artifacts* feed to public repositories (e.g., npmjs.com, nuget.org) or other *Azure Artifacts* feeds. This allows packages from upstream sources to be cached and consumed via your feed.