Mobile Application Development
Aileen Pierce

PERSISTENT DATA

Persistent Data

- Shared Preferences
- Files
  - Internal storage
  - External storage
- SQL Database
- Network Connection

Shared Preferences

- You can save a small amount of data as key-value sets using the **`SharedPreferences`** API
- Use **`getPreferences()`** if you're only using one shared preference file as this uses the default file name
- Use **`getSharedPreferences()`** if you need multiple shared preferences files each with a unique name

Shared Preferences

- You can save a small amount of data as key-value sets using the **`SharedPreferences`** API
- Writing to a shared preferences file

1. create a **`SharedPreferences.Editor`** by calling **`edit()`** on your SharedPreferences
2. Add the key-value pairs using methods like **`putInt()`**, **`putString()`**, and **`putStringSet()`**
3. Call **`commit()`** to save the changes

Shared Preferences

- Reading from a shared preferences file

1. create a **`SharedPreferences.Editor`** by calling **`edit()`** on your SharedPreferences
2. Read in the key-value pairs using methods like **`getInt()`**, **`getString()`**, and **`getStringSet()`**

Internal Storage

- Internal storage should be used for a larger amount on unstructured data private to your app
- Always available
- By default files saved are private to your app
- Other apps and the user can't access the files
- Files are removed when the user uninstalls your app
- Internal storage is best when you want to be sure that neither the user or other apps can access your files

External Storage

- Similar to internal storage but the data is world-readable
- Only available when the storage is accessible
- Files can be read outside the app and out of your control
- When the user uninstalls the app the files are removed only if you saved them in the directory **`getExternalFilesDir()`**
- External storage is best for files that don't require access permissions or that you want to share with other apps or users

SQL Database

- A SQL database is a good choice for a large amount of structured data
- Android includes the APIs you need to use a SQL database in the **`android.database.sqlite`** package
- Using a database is a good choice when you have structured data

Network Connection

- You can use the network, when it's available, to store and retrieve data on your own web-based services
- To use network operations use the **`java.net.*`** and **`android.net.*`** classes