Update example dependencies

Follow these steps to update the examples to the latest versions of Angular (and related dependencies):

- In shared/package.json, bump all the `@angular/*` and `@nguniversal/*` package versions to the version you want to update to and update their peer dependencies (such as `@angular-devkit/*`, `rxjs`, `typescript`, `zone.js`) and other dependencies (e.g. `@types/*`) to the latest compatible versions.

  > NOTE: The angular-cli-diff repo can be a useful resource for discovering what dependency versions are used for a basic CLI app at a specific CLI version.

- In the shared/ directory, run `yarn` to update the dependencies in the shared/node_modules/ directory and the shared/yarn.lock file.
- In the shared/ directory, run `yarn sync-deps` to update the dependency versions of the `package.json` files in each sub-folder of shared/boilerplate/ to match the ones in shared/package.json.
- Follow the steps in the following section to update the rest of the boilerplate files.

Update other boilerplate files

The Angular CLI default setup is updated using `ng update`. Any necessary changes to boilerplate files will be done automatically through migration schematics.

> NOTE: Migrations affecting source code files will not happen automatically, because `ng update` does not know about all the examples in `aio/content/examples/`. You have to make these changes (if any) manually. Again, the angular-cli-diff repo can be a useful resource for discovering changes between versions.

- In the shared/boilerplate/cli/ directory, run the following commands to migrate the project to the current versions of Angular CLI and the Angular framework (updated in previous steps):

  ```
  # Ensure dependencies are installed.
  yarn install

  # Migrate project to new versions.
  yarn ng update @angular/cli --allow-dirty --migrate-only --from=<previous-cli-version>
  yarn ng update @angular/core --allow-dirty --migrate-only --from=<previous-core-version
  ```

  > NOTE: In order for `ng update` to work, there must be a `node_modules/` directory with installed dependencies inside the shared/boilerplate/cli/ directory. This `node_modules/`

directory is only needed during the update operation and is otherwise ignored (both by git and by the example-boilerplate.js script) by means of the shared/boilerplate/.gitignore file.

- The previous command made any necessary changes to boilerplate files inside the `cli/` directory, but the same changes need to be applied to the other CLI-based boilerplate directories. Inspect the changes in `cli/` and manually apply the necessary ones to other CLI-based boilerplate directories.
- Also ensure that any relevant changes in the shared/boilerplate/cli/ directory are copied to the shared/example-scaffold/ directory, which is used when creating new examples (via `yarn create-example ...`). Only files that would not be considered boilerplate should be added to the `example-scaffold/` directory.
- Run the following command to list all the boilerplate files that are overridden in specific examples.

  ```
  node aio/tools/examples/example-boilerplate.js list-overrides
  ```

  Inspect each of these files to determine whether they need to be updated.
- Inspect the changes and determine whether some of them need to be applied to the `systemjs` boilerplate files.
- Inspect the changes and determine whether any updates to guides are necessary. For example, if a file is renamed or moved, any guides mentioning that file may need updating to refer to the new name/location.
- Commit all changes to the repository.