z_make_gitignore_sync_tool_x

Synchronise a single, canonical `.gitignore` across every repository in the `x_runner_x` workspace. The tool reads a shared template, ensures each repo has the exact same contents, and reports what changed.

Why this exists

- Git caches, reports and build artefacts were sneaking into different repos.
- Shared tooling (pytest, mypy, Ruff, etc.) needs consistent exclusions.
- New repositories should inherit the template automatically without copy/paste.

What the tool does

1. Loads the canonical template from `resources/gitignore-template.txt`.
2. Discovers every sibling repository (folders containing a `.git` directory) beneath the workspace root.
3. Compares each repoâ€™s `.gitignore` to the template and overwrites it when any difference is found (or creates the file when missing).
4. Prints a short summary of updates. A `--dry-run` flag shows pending changes without touching disk.

The operation is idempotent: running it repeatedly leaves repositories unchanged once they already match the template.

Usage

```powershell

# from the workspace root (c:\x_runner_x)

pwsh -File scripts\run_gitignore_sync.ps1

# or directly via Python

C:\x_runner_x.venv\Scripts\python.exe -m x_make_gitignore_sync_x.sync

# helpful flags

python -m x_make_gitignore_sync_x.sync --dry-run python -m x_make_gitignore_sync_x.sync --root C:\other\workspace python -m x_make_gitignore_sync_x.sync --template custom-template.txt ```

## Integration points

- `z_make_all_x` should invoke this tool near the end of its flow once all repos have been cloned or refreshed. That guarantees existing and freshly created repositories converge on the shared template.
- New repo generators (e.g. scaffolding commands) can either call the sync tool or copy the template directly during bootstrap.

## Tests

See `tests/test_sync.py` for coverage of the repo discovery logic, template application, and dry-run behaviour.
```