AngularJS Release instructions

Compare the list of commits between stable and unstable

There is a script - compare-master-to-stable.js - that helps with this. We just want to make sure that good commits (low risk fixes + docs fixes) got cherry-picked into stable branch and nothing interesting got merged only into stable branch.

Pick a release name (for this version)

A super-heroic power (adverb-verb phrase).

Generate release notes

Example Commit: https://github.com/angular/angular.js/commit/7ab5098c14ee4f195dbfe2681e402fe2dfeacd78

1. Run

```
node_modules/.bin/changez -o changes.md -v <new version> <base branch>
```

2. Review the generated file and manually fix typos, group and reorder stuff if needed.
3. Move the content into CHANGELOG.md add release code-names to headers.
4. Push the changes to your private github repo and review.
5. cherry-pick the release notes commit to the appropriate branches.

Pick a commit to release (for this version)

Usually this will be the commit containing the release notes, but it may also be in the past.

Run "release" script

```
scripts/release/release.sh --git-push-dryrun=false --commit-sha=8822a4f --version-number=1.7.6 --version-name=gravity-manipulation
```

1. The SHA is of the commit to release (could be in the past).
2. The version number and code-name that should be released, not the next version number (e.g. to release 1.2.12 you enter 1.2.12 as release version and the code-name that was picked for 1.2.12, cauliflower-eradication).
3. You will need to have write access to all the AngularJS github dist repositories and publish rights for the AngularJS packages on npm.

Update GitHub milestones

1. Create the next milestone if it doesn't exist yet-giving ita due date.
2. Move all open issues and PRs for the current milestone to the next milestone
   You can do this by filtering the current milestone, selecting via checklist, and moving to the next milestone within the GH issues page.

3. Close the current milestone click the milestones tab and close from there.

4. Create a new holding milestone for the release after next-but don't give it a due date otherwise that will mess up the dashboard.

Push build artifacts to CDN

Google CDNs are fed with data from google3 every day at 11:15am PT it takes only few minutes for the import to propagate). If we want to make our files available, we need submit our CLs before this time on the day of the release.

Don't update the package.json (branchVersion) until the CDN has updated

This is the version used to compute what version to link to in the CDN. If you update this too early then the CDN lookup fails and you end up with 'null, for the version, which breaks the docs.

Verify angularjs.org download modal has latest version (updates via CI job)

The versions in the modal are updated (based on the versions available on CDN) as part of the CI deploy stage. (You may need to explicitly trigger the CI job. e.g. re-running the last `deploy` job.)

Announce the release (via official Google accounts)

Double check that angularjs.org is up to date with the new release version before sharing.

1. Collect a list of contributors

use: `git log --format='%aN' v1.2.12..v1.2.13 | sort -u`

2. Write a blog post (for minor releases, not patch releases) and publish it with the "release" tag
3. Post on twitter as yourself (tweet from your heart; there is no template for this), retweet as @AngularJS

Party!

Major Release Tasks

1. Update angularjs.org to use the latest branch.
2. Write up a migration document.
3. Create a new git branch for the version that has been released (e.g. 1.8.x).
4. Check that the build and release scripts still work.
5. Update the dist-tag of the old branch, see https://github.com/angular/angular.js/pull/12722.
6. Write a blog post.