THE AGILE DBA: DATABASE DEVOPS WITH SSDT

Ryan Booz

VP of BI & CTO
EnergyCAP, Inc.

@RyanBooz

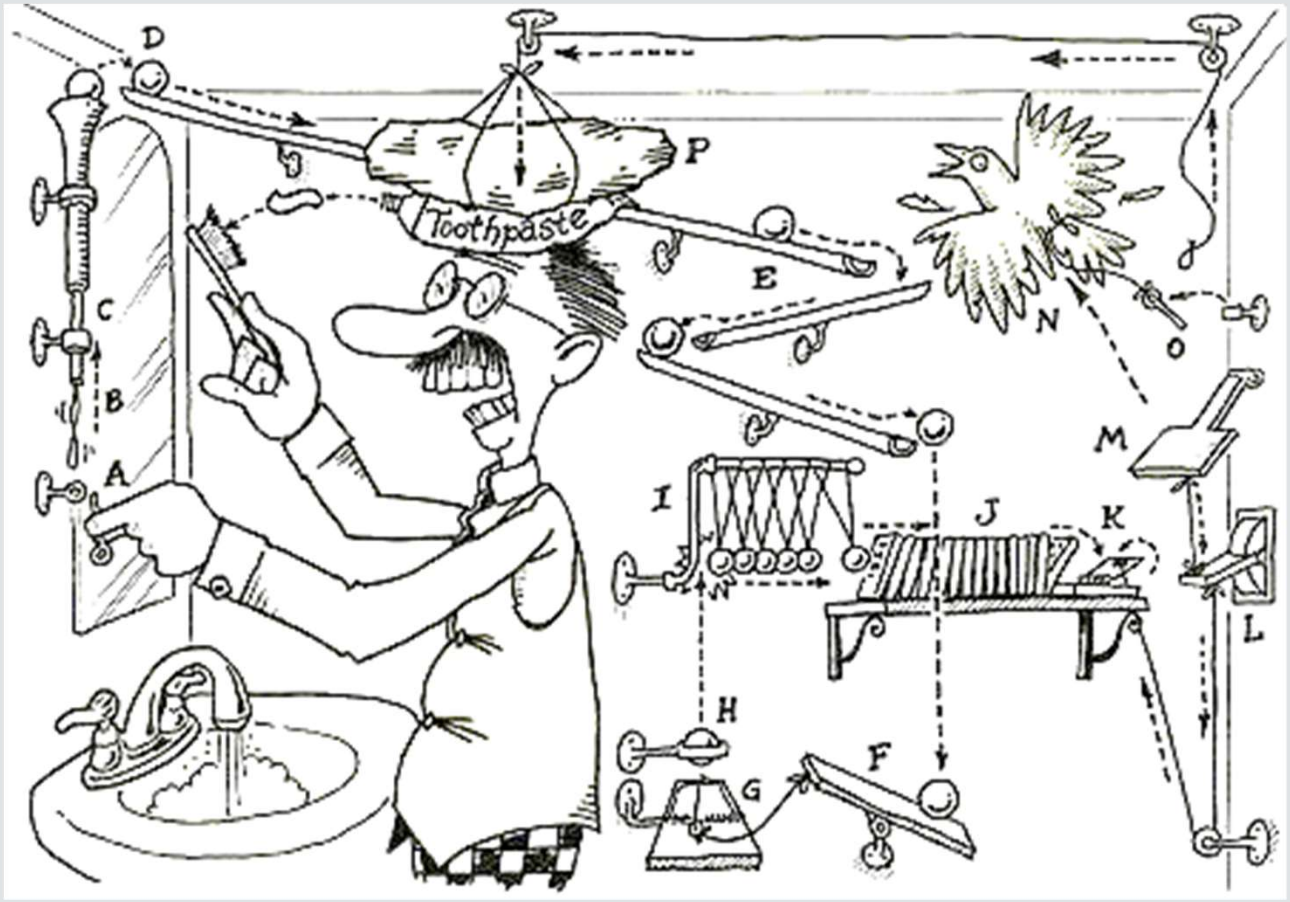
www.SoftwareAndBooz.com

ABOUT ME

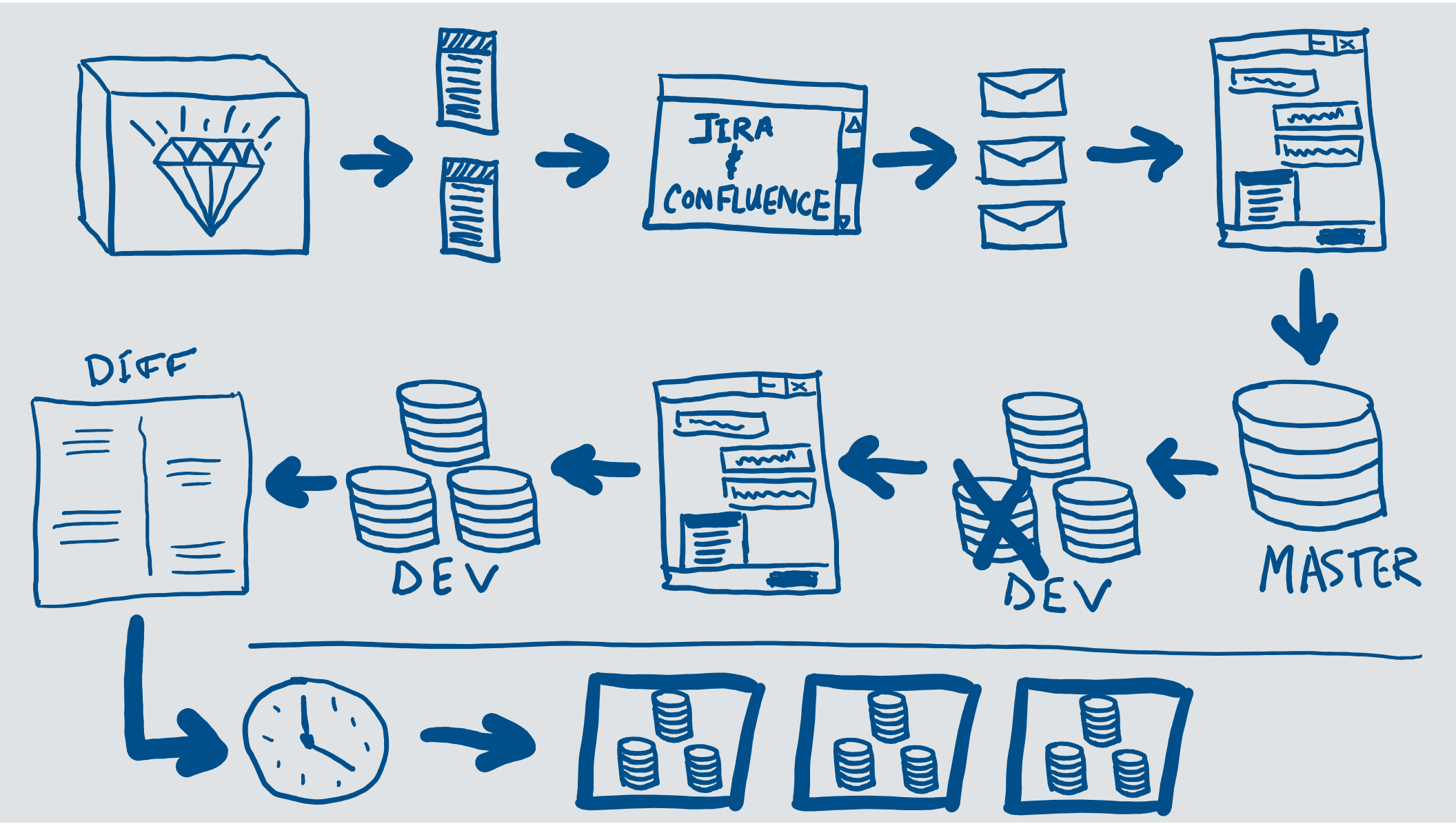
- Husband and Father of 5… soon to be 6 ☺
- Own 33 acres,10 chickens & 2 beehives
- 18 years of Relational DB experience – even DB2!!
- Learned to program on a DEC 5000 in high school with the Pittsburgh Supercomputing Center
- Started in CompSci, graduated with Music Ed

AGENDA

- What Problem are we solving?
- Overview of SSDT
- Creating a Project from an existing database
- Develop and Deploy Locally
- Continuous Integration & Deployment in Dev
- What about Production?
- A handful of gotchas
- Resources

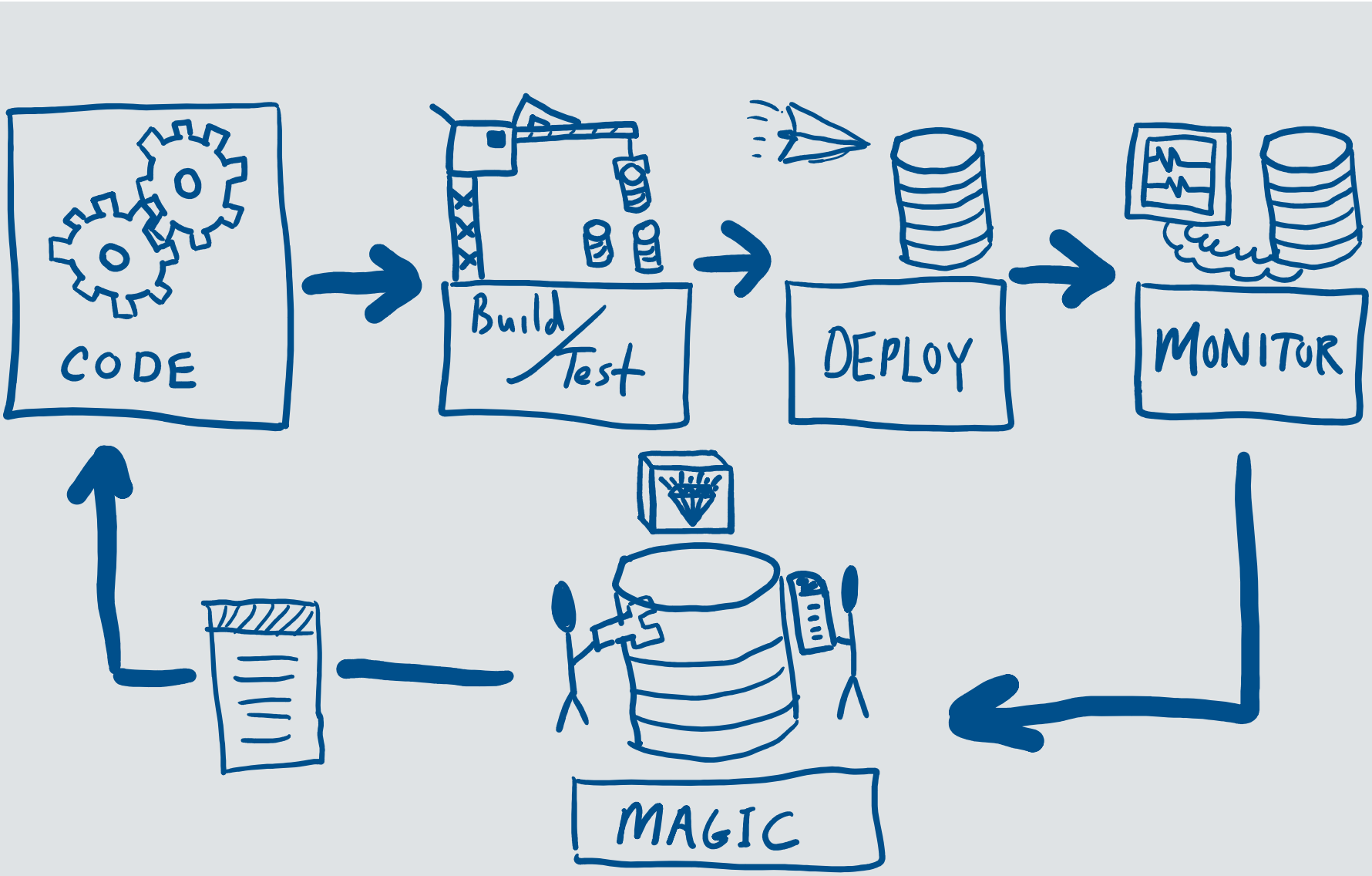


THE OLD WAY


JIRA
&
CONFLUENCE
MASTER
DEV
DEV
DIFF

THE DEVOPS WAY

http://potus98.wikidot.com/wiki:devopsdays2012


CODE
Build
Test
DEPLOY
MONITOR
MAGIC

GITFLOW

MORE DEVOPS INFORMATION

Check out these other resources:

- https://groupby.org/conference-session-abstracts/bringing-devops-to-the-database/
- https://groupby.org/conference-session-abstracts/devops-101-for-data-professionals-how-your-jobs-will-change/
- The Phoenix Project (book)
- The DevOps Handbook (book)

SSDT OVERVIEW

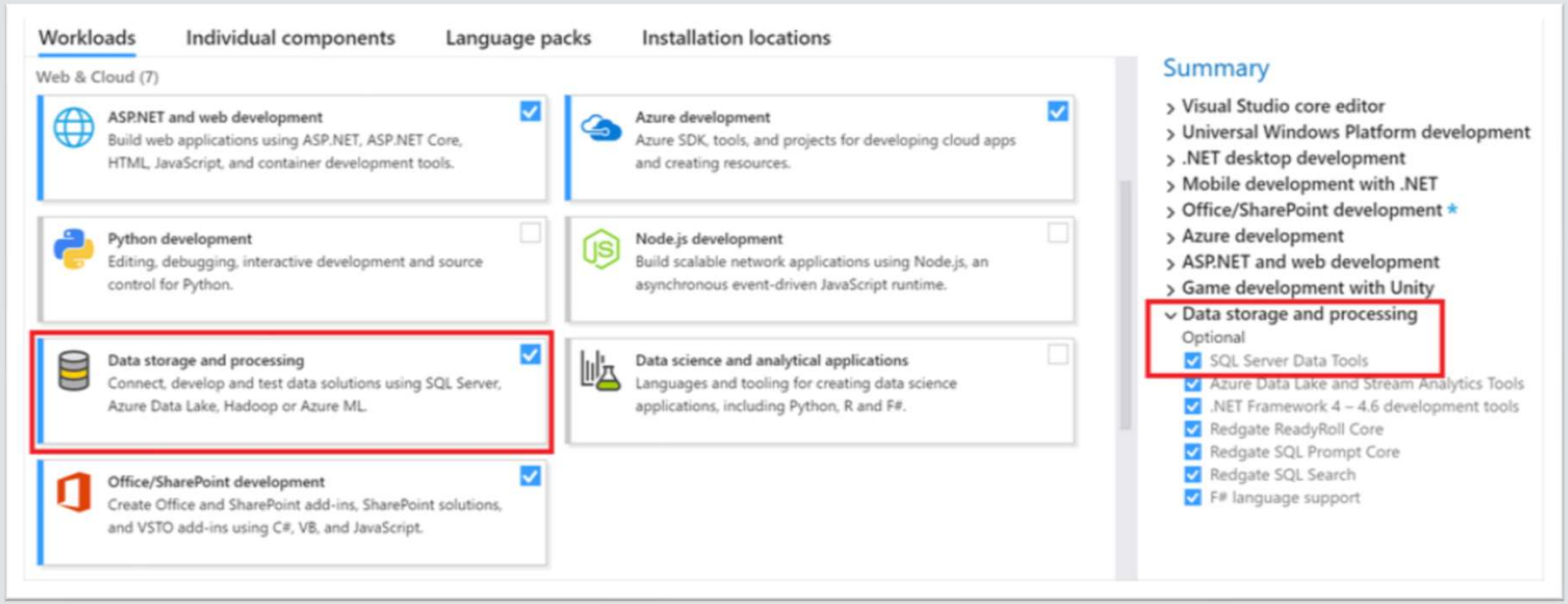
WHY SSDT?

- Declarative, not Migration (Redgate ReadyRoll)
  - DACPAC is the key here
- Rich exploration of database as code
- Validation of Schema and Objects
- Code Analysis
- Refactoring
- Database Schema Comparison
- Consistent, fine-grained deployment control

PREREQUISITS

- Visual Studio 2017 (any edition)
- A Build environment (Visual Studio Team Services)
- Local version of SQL Server (SQL 2017 Developer)
- Source control (again, VSTS can provide this)
  - Gitflow or similar branching definition

https://docs.microsoft.com/en-us/sql/ssdt/download-sql-server-data-tools-ssdt



DATA PROJECTS COMPONENTS

DACPAC	• Glorified ZIP file with model XML and other meta data
MSBUILD	• Same build tooling as other development projects • Creates DACPAC from Database Project declared model
SQLPackage	• Compares two DACPAC files to generate differences • Create change scripts, apply changes, generate deployment reports • Configure through publish profiles and command-line • Deployment contributors provide endless opportunities • Cross-Platform

MISSION: FIND A NEW BABY NAME!

- Use Kendra Little's BabbyNames database
  - https://github.com/LitKnd/BabbyNames
- Apply Trigram search ability based on POC from Paul White
  - https://sqlperformance.com/2017/09/sql-performance/sql-server-trigram-wildcard-search
- Find names similar to the one we are searching for, most popular first

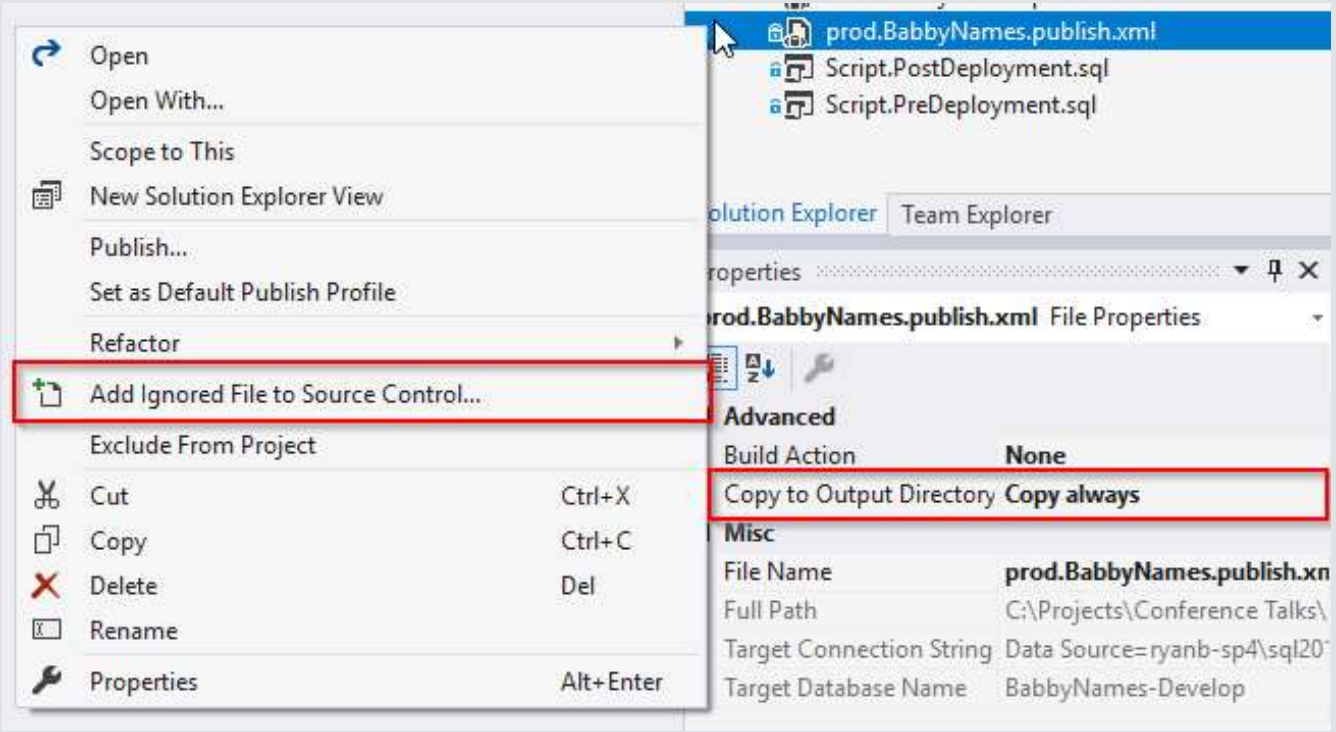
SSDT DEMO

MINIMUM PROFILE SETTINGS

- At least consider these for each profile:
  - Generate smart defaults
  - Ignore column order
  - Automatically take a backup
  - Include Transactional Scripts
  - Allow Incompatible Platform – between version deployments
- Never select "Always Re-create Database"
  - Just sayin'

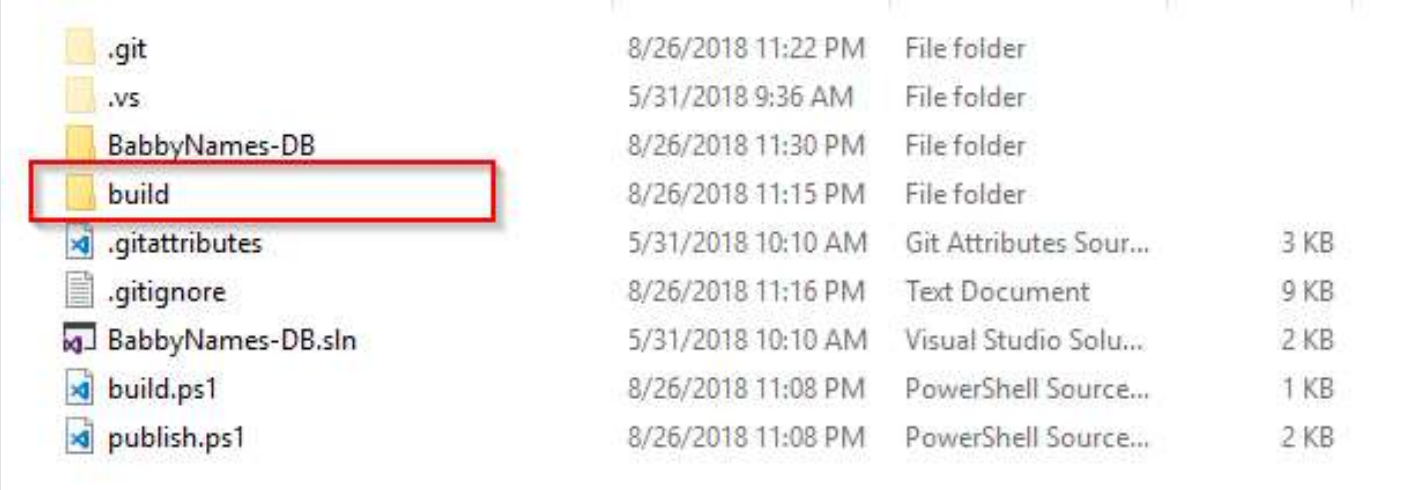
MINIMUM DEPLOYMENT STEPS

1. Publish Profile for each environment/type



MINIMUM DEPLOYMENT STEPS

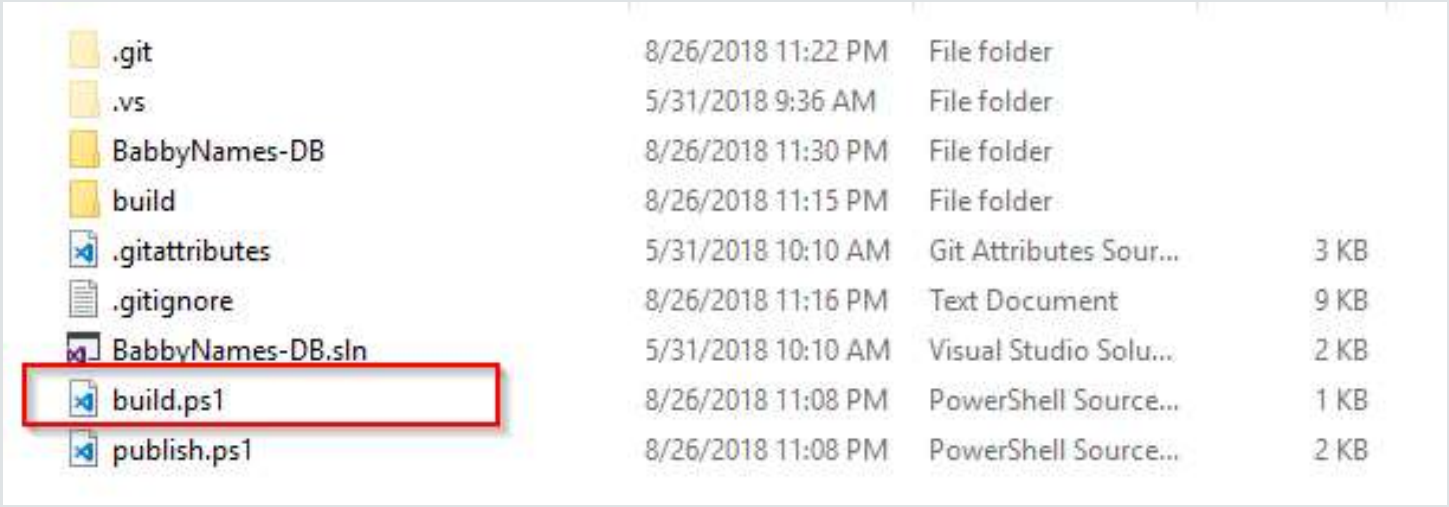
2. MSBuild Tools installed in Solution Folder (not included in solution definition)



Install into folder from: https://www.nuget.org/packages/Microsoft.Data.Tools.Msbuild/

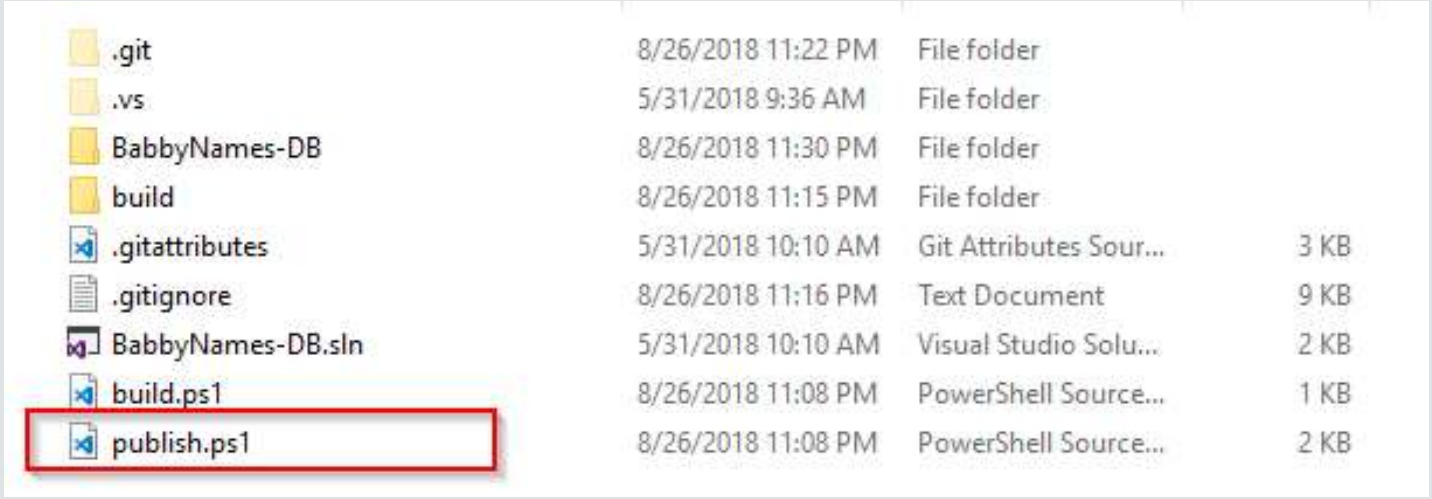
MINIMUM DEPLOYMENT STEPS

3. Build Script (shown at root of Solution)

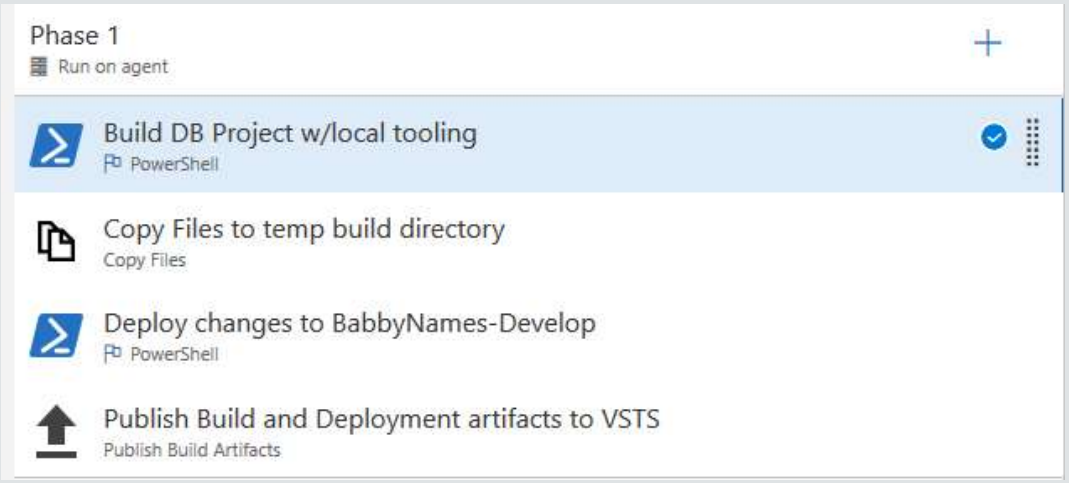


MINIMUM DEPLOYMENT STEPS

4. Publish Script (shown at root of Solution)

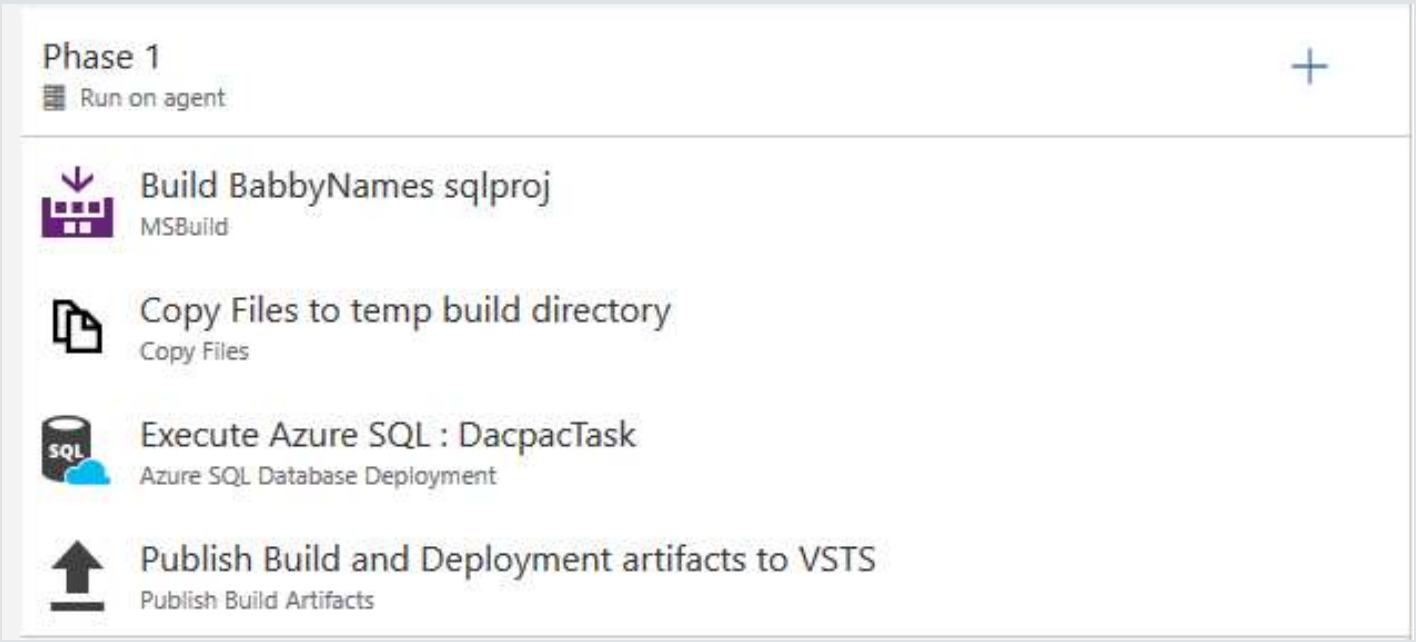


BUILD AND PUBLISH – ON-PREM



- Install and configure local build agent
- Install Microsoft.Data.Tools.Msbuild NuGet package locally
- Create Powershell script to run the build
- Create Powershell script to run SqlPackage to Publish the database

BUILD AND PUBLISH - AZURE



BUILD & DEPLOY DEMO

DEPLOYMENT CONTRIBUTORS

GAME CHANGING FEATURE

- Examine and modify DB Model
- Deployment-time decisions with access to script, source and target model
- Provides significant flexibility for well understood needs
- Steep learning curve
- But come on… it's fun!

DEPLOYMENT CONTRIBUTOR DEMO

WHAT ABOUT PRODUCTION?

- Depends how automated you want to be and risk tolerance
  - Lots of DBs might not scale based on maintenance windows
  - Can be transactional, which should minimize some risk aside from wasted time
- Deployment Contributors
- "Generate Script" against previous version of database
  - Allows final review of scripts
  - Any additional edits that might not be feasible in SSDT
  - Provide to clients
  - Use familiar tooling - SSMS

GOTCHAS

- MSBuild for Database Projects is not Cross-Platform yet (Windows only)
- Inconsistencies between your code and how SQL stores it, causing the same upgrade every time:
  - Data Motion
  - Check constraint (IN vs OR)
  - Fill Factor isn't always ignored
  - Permissions out of Sync
- Using References is not trivial
- Order of upgrade script is determined by SQLPackage, which might not always make preferable decisions
- No concept of "Rollback", "Roll-forward" instead

FURTHER READING

- SSDT Docs: https://msdn.microsoft.com/en-us/library/hh272686(v=vs.103).aspx
- SqlPackage Docs: https://msdn.microsoft.com/library/hh550080(vs.103).aspx
- Deployment Contributors: https://msdn.microsoft.com/en-us/library/dn306642(v=vs.103).aspx
- Azure CI/CD Docs: https://blogs.msdn.microsoft.com/ssdt/2016/04/06/sqldb-cicd-intro/
- Ed Elliot: https://the.agilesql.club/

THANK YOU!