inQdo

WHITE PAPER.

Projen: De volgende stap in projectconfiguratie

Hoe automatische projectstructuren leiden tot efficiëntere ontwikkelteams en consistentere codebases

Maart 2025

INHOUD.

Inleiding

“Even een nieuw project opzetten.” Die ogenschijnlijk simpele taak groeit in moderne ontwikkelteams niet zelden uit tot een complexe uitdaging. Niet omdat het starten van een project zelf zo ingewikkeld is, maar omdat de wildgroei aan verschillende configuraties, tools en best practices het consistent houden van projecten steeds lastiger maakt.

Projectconfiguratie is een term die je zelden zult horen buiten technische kringen, maar binnen softwareontwikkeling is het een cruciaal onderdeel van elk succesvol project. Het vormt het fundament waarop software wordt gebouwd en onderhouden, en de kwaliteit ervan heeft directe impact op de efficiëntie en samenwerking binnen teams. In kleinere teams met slechts een handvol projecten blijft deze taak vaak beheersbaar. Maar wat als je werkt met tientallen of zelfs honderden projecten die allemaal hun eigen configuraties nodig hebben?

Voor ontwikkelteams die werken met meer dan 200 repositories is dit geen hypothetisch scenario, maar dagelijkse realiteit. Elk project binnen die repositories vereist unieke configuratiebestanden zoals *package.json, tsconfig.json, .eslintrc.json, .prettierrc, .gitignore* en CI/CD-workflows. Het aanpassen van deze bestanden gebeurt vaak handmatig, wat niet alleen tijdrovend is, maar ook de kans op menselijke fouten vergroot. Een kleine aanpassing in de linting-regels betekent handmatig tientallen of zelfs honderden repositories updaten. Een nieuwe security-best-practice implementeren? Die exercitie kost dagen of zelfs weken.

Dit probleem wordt alleen maar groter naarmate teams groeien en projecten complexer worden.

Dit probleem wordt alleen maar groter naarmate teams groeien en projecten complexer worden. Inconsistente configuraties kunnen leiden tot:

- Tijdrovende code reviews die zich richten op stijl- en formatkwesties in plaats van functionaliteit.
- Verhoogde technische schuld, omdat handmatige fixes vaak ad hoc worden toegepast.

- Frustratie binnen teams, omdat het gebrek aan standaardisatie de onboarding van nieuwe teamleden bemoeilijkt.

Traditionele tools versus Projen

Traditionele projectconfiguratietools zoals Cookiecutter en GitHub templates hebben geprobeerd deze uitdagingen aan te pakken. Ze bieden vaak een goede start door een projecttemplate te genereren, maar daar houdt het op. Zodra het project draait, is het aan het team om handmatig wijzigingen door te voeren en bij te houden. Ook CDK-init kent deze beperking.

Dit leidt tot situaties waarin configuratiebestanden al snel uit de pas lopen

Dit leidt tot situaties waarin configuratiebestanden al snel uit de pas lopen. Een template die ooit alle benodigde onderdelen bevatte, wordt maanden later irrelevant omdat best practices zijn veranderd. Dit maakt het handhaven van consistentie bijna onmogelijk.

Hier komt Projen in beeld. Projen is niet zomaar een tool; het is een filosofie. Het behandelt configuratie als code, wat betekent dat projectregels en instellingen centraal worden beheerd en geautomatiseerd. Door een enkele bron van waarheid te creëren in de vorm van een projenrc.ts-bestand, zorgt Projen voor consistente, up-to-date configuraties in elk project.

Wat Projen onderscheidt van de andere tools, is de combinatie van eenvoud en kracht. Het biedt niet alleen een gestroomlijnde manier om projecten te starten, maar ook een mechanisme om ze continu te beheren en bij te werken. Met Projen hoef je je geen zorgen meer te maken over handmatige fixes of vergeten updates. Alles wordt automatisch gegenereerd en beheerd op basis van de nieuwste standaarden.

1

Van CDK naar Projen: Een natuurlijke evolutie

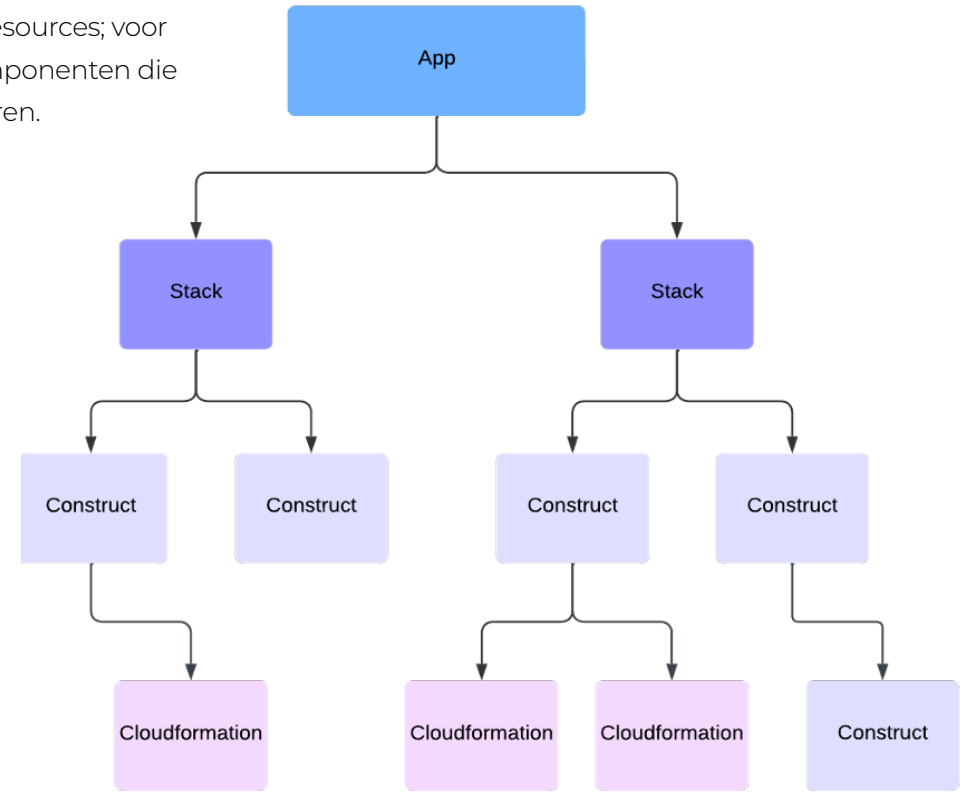
Om de kracht van Projen te begrijpen, helpt het om te kijken naar een concept dat veel ontwikkelteams al kennen: AWS CDK (Cloud Development Kit). Beide tools zijn door dezelfde ontwikkelaar bedacht: Elad Ben-Israel. Zowel Projen als AWS CDK maakt gebruik van de 'construct' library die een modulaire hiërarchie mogelijk maakt. Ook wordt er gebruikgemaakt van JSII, waardoor project templates ook in andere programmeertalen kunnen worden gebruikt. Dit stelt teams in staat om op een hoger abstractieniveau te werken, terwijl technische details automatisch worden afgehandeld.

De parallel tussen CDK en Projen is treffend. Beide tools abstraheren complexe processen en werken met een modulaire hiërarchie die aanpasbaar en schaalbaar is. Voor CDK zijn dit constructen die leiden tot CloudFormation-resources; voor Projen zijn het componenten die bestanden genereren.

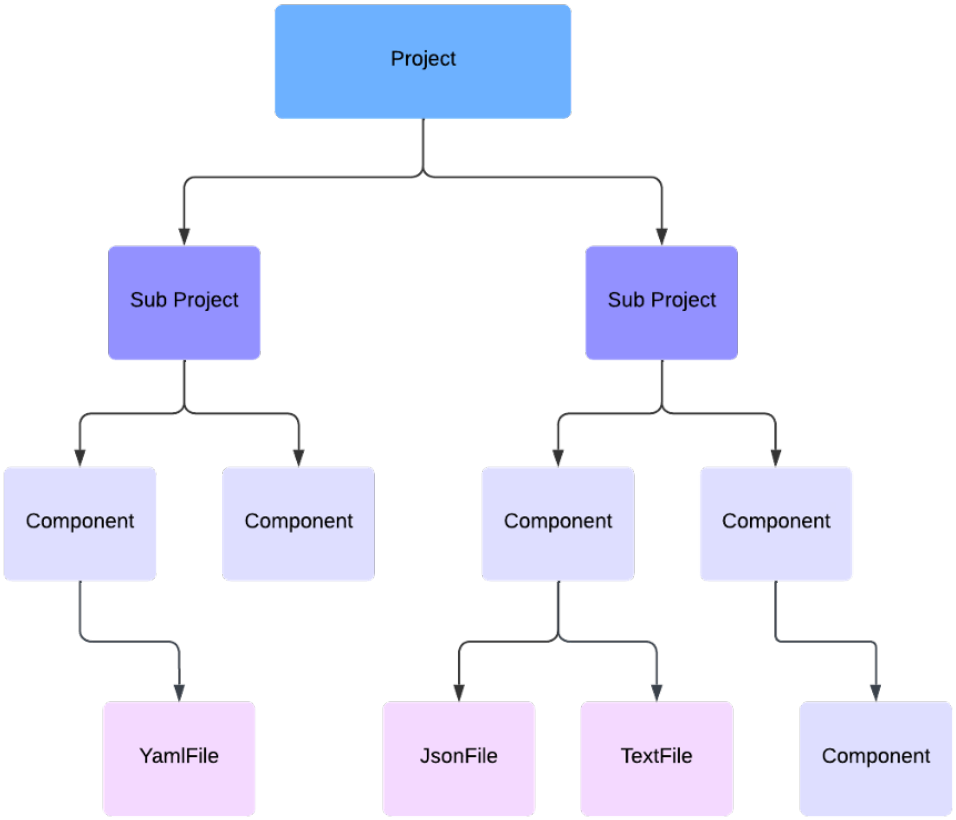
AWS-CDK

AWS CDK werkt met een duidelijke hiërarchie:

- App vormt de basis en bevat één of meer stacks
- Stack bevat één of meer constructs
- Constructs leiden uiteindelijk tot concrete CloudFormation resources



Figuur 2.1 > Schematische weergave van AWS CDK



< Figuur 2.2
Schematische weergave van Projen

Projen

Projen volgt een vergelijkbaar concept:

- Project vormt de basis en kan sub-projecten bevatten
- Sub-projecten hebben componenten
- Componenten genereren uiteindelijk concrete configuratiebestanden

Deze parallel is geen toeval. Net zoals CDK ontwikkelaars bevrijdt van het handmatig schrijven van CloudFormation templates, bevrijdt Projen teams van het handmatig beheren van configuratiebestanden. Het verheft projectconfiguratie naar een hoger niveau van abstractie, waar het makkelijker te beheren en te onderhouden is. Bij elk project dat met Projen wordt gestart, wordt een centraal *projenrc.ts*-bestand gegenereerd. Hierin wordt de volledige projectconfiguratie gedefinieerd, van afhankelijkheden en linting-regels tot build-scripts en workflows.

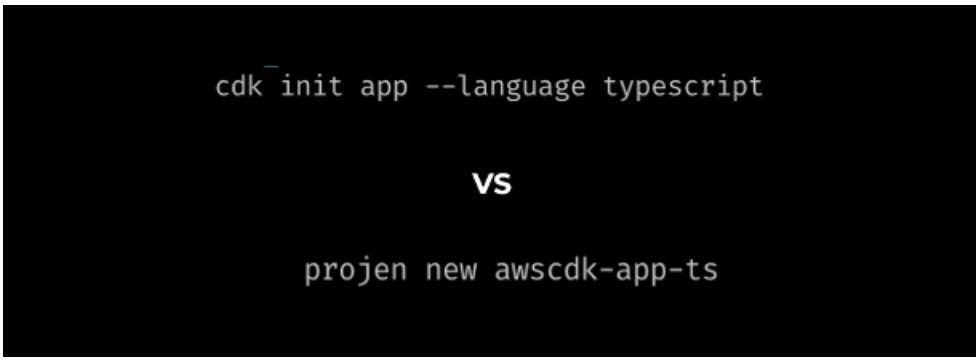
De praktische impact

In de praktijk lost deze aanpak direct een aantal hardnekkige problemen op die veel ontwikkelteams ervaren:

- **Consistentie over projecten heen**
  Het onderhouden van meer dan 200 repositories betekent dat zelfs kleine wijzigingen, zoals het aanpassen van een linting-regel, een enorme impact kunnen hebben. Door configuratie als code te behandelen, worden dit soort updates een kwestie van minuten in plaats van dagen of weken.

- **Eenvoudigere samenwerking**
  Code reviews worden effectiever omdat teams zich kunnen concentreren op functionaliteit in plaats van op formattering of configuratie. Een pull request toont alleen de relevante wijzigingen, niet de ruis van inconsistente configuraties.

- **Snellere onboarding**
  Nieuwe teamleden, of ontwikkelaars die tijdelijk bij een project aansluiten, kunnen direct aan de slag. Met één commando hebben ze een volledig geconfigureerde ontwikkelomgeving die voldoet aan alle teamstandaarden:



< Figuur 2.3
Verschil in aanroepprompt tussen CDK en Projen

- **Geautomatiseerde veiligheid**
  Security-tools zoals cdk-nag kunnen standaard worden geïntegreerd. Dit zorgt ervoor dat beveiligingsrichtlijnen consequent worden toegepast in alle projecten.

2

Projen in de praktijk

Projen is meer dan een theoretisch concept; het is een tool die bewezen heeft processen te stroomlijnen, tijd te besparen en de kwaliteit van werk te verbeteren. Dit wordt duidelijk in de manier waarop teams, zoals inQdo, het inzetten om dagelijkse uitdagingen op te lossen.

Een van de grootste krachten van Projen is de mogelijkheid om configuratiebestanden automatisch te genereren en te onderhouden. Waar ontwikkelaars traditioneel veel tijd kwijt zijn aan het handmatig aanmaken en beheren van projectconfiguraties, biedt Projen een geautomatiseerde oplossing die niet alleen tijd bespaart, maar ook consistentie garandeert.

Neem bijvoorbeeld het opzetten van een CDK-project. Zonder Projen zou een ontwikkelaar een reeks bestanden handmatig moeten aanmaken en configureren. Denk aan essentiële bestanden zoals *.gitignore, package.json, en tsconfig.json*, maar ook complexere onderdelen zoals CI/CD-workflows voor GitHub Actions. Het proces vereist zorgvuldigheid, want een kleine fout kan leiden tot inconsistente configuraties of problemen bij de implementatie.

Met Projen wordt deze complexe taak teruggebracht tot een enkele opdracht. Door simpelweg het volgende commando uit te voeren:

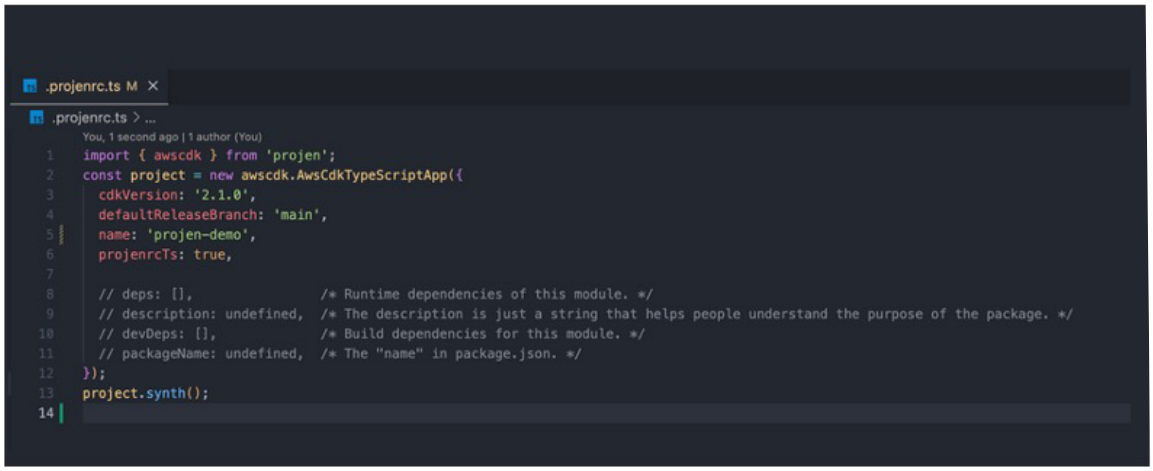
projen new awscdk-app-ts

wordt automatisch een volledig geconfigureerde projectomgeving gegenereerd. Binnen enkele seconden beschikt de ontwikkelaar over een projectstructuur die voldoet aan de vooraf gedefinieerde standaarden van het team.

De gegenereerde projectstructuur ziet er bijvoorbeeld zo uit:

my-project/
- ***.gitignore***
- ***package.json***
- ***tsconfig.json***
- ***.eslintrc.json***
- ***.projen/***
- ***workflows/***
- ***src/***

Projenrc.ts



< Figuur 3.1
De centrale configuratie binnen Projen: Projenrc.ts

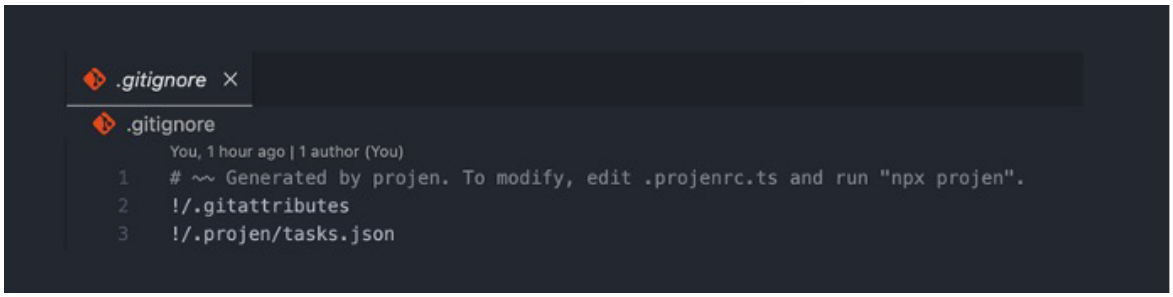
Elk bestand is automatisch afgestemd op de gestelde standaarden, van linting- en formatteringregels tot aan de workflow-configuraties. Wat voorheen een tijdrovend en foutgevoelig proces was, is nu een efficiënte, gestroomlijnde stap geworden in de dagelijkse praktijk van ontwikkelteams.

Projen geeft ontwikkelteams niet alleen tijd terug, maar zorgt ook voor een solide basis waarop verder gebouwd kan worden. Door configuraties centraal te beheren en automatisch up-to-date te houden, krijgen ontwikkelaars de vrijheid om zich te richten op wat echt belangrijk is: het bouwen van innovatieve oplossingen [zie ook volgend hoofdstuk].

Managed versus unmanaged bestanden

Wat Projen echt onderscheidt van andere oplossingen is het concept van ‘managed’ en ‘unmanaged’ bestanden. Dit fundamentele principe bepaalt hoe configuraties worden beheerd en up-to-date blijven.

Managed files can be identified by the Projen marker



< Figuur 3.2
Managed files worden door Projen beheerd

Managed bestanden:
Automatisch en consistent

Managed bestanden worden volledig door Projen beheerd en bij elke update opnieuw gegenereerd. Denk aan configuratiebestanden voor:

- Linting en code formatting
- Build processen
- CI/CD workflows
- Dependency management

Deze bestanden zijn herkenbaar aan een speciale Projen-markering die aangeeft dat ze automatisch worden beheerd. Handmatige wijzigingen in deze bestanden worden overschreven - een bewuste keuze die consistentie garandeert.

Unmanaged bestanden:
Flexibiliteit waar nodig

Niet alle bestanden hebben strikte controle nodig. Unmanaged bestanden worden eenmalig gegenereerd als startpunt en kunnen daarna vrij worden aangepast. Dit is ideaal voor:

- Voorbeeldcode en templates
- Projectspecifieke implementaties
- Documentatie
- Custom configuraties

Voorbeeld:
Stel dat je een *.gitignore*-bestand hebt waarin regels staan voor het negeren van bestanden die naar Version Control gepushed worden. Projen zorgt ervoor dat deze regels altijd up-to-date blijven volgens de teamstandaarden. Handmatige toevoegingen aan het bestand worden genegeerd, maar deze dienen via de projenrc.ts toegevoegd te worden.

3

Configuratie configureren

Wat maakt Projen anders?

De fundamentele beperking van traditionele oplossingen is dat ze configuratiebeheer behandelen als een eenmalige actie - het opzetten van een project - in plaats van als een continu proces. In de praktijk evolueert een development team constant. Nieuwe security-patches moeten worden toegepast, coding standards worden aangescherpt, dependencies moeten worden geüpdatet. Zonder geautomatiseerd beheer wordt dit een bijna onmogelijke taak.

> Het verschil met Projen wordt duidelijk als we kijken naar hoe teams omgaan met bijvoorbeeld een nieuwe security-best-practice. Met traditionele oplossingen zou dit betekenen:
>
> 1. De template of het script updaten voor nieuwe projecten
> 2. Een inventarisatie maken van alle bestaande projecten die aangepast moeten worden
> 3. Handmatig alle repositories langsgaan om de wijziging door te voeren
> 4. Code reviews uitvoeren op elke wijziging
> 5. Hopen dat geen enkel project over het hoofd is gezien

Met Projen daarentegen wordt de nieuwe practice toegevoegd aan de centrale configuratie. Doordat het Configuratie-as-Code is, wordt de configuratie van de configuratie beheerbaar in version control zoals Git/GitHub. Daardoor ontstaat er een geschiedenis van changes, en een overzicht wie wat heeft aangepast. Bovendien wordt het project template beschikbaar gesteld als een ‘package’.

Wanneer een nieuw project wordt gestart, kan de package worden gedownload en heeft het nieuwe project een afhankelijkheid op deze package. Dat betekent dat wanneer de inhoud van de package verandert (de configuratie), het project automatisch wordt geüpdatet naar de nieuwste versie van het project template door de package te updaten. Dit gebeurt met 1 commando (npm update), waarbij alle veranderingen doorgevoerd worden. Het uitvoeren van package updates kan bovendien ook weer geautomatiseerd worden.

Projen is ontworpen om de beperkingen van de traditionele tools te overwinnen. Wijzigingen worden automatisch en consistent doorgevoerd, zonder handmatig werk en met minimale kans op fouten. Het gaat verder dan het genereren van een initiële setup en biedt een continu beheermodel.

Traditionele tools	Projen
Focust op initiële projectsetup	Beheert projecten gedurende hun hele levenscyclus
Handmatige updates voor configuraties	Automatische updates via managed bestanden
Beperkt tot éénmalige sjablonen	Gebruik van herbruikbare componenten
Geen geïntegreerd beheer van afhankelijkheden	Volledig afhankelijkheidsbeheer binnen projenrc.ts

4

Van theorie naar praktijk: Projen implementeren

De stap naar Projen vraagt om een doordachte aanpak. De realiteit van een ontwikkelteam met meer dan 200 repositories is dat je niet alles in één keer kunt omgooien. Ingesleten werkwijzen, bestaande configuraties en lopende projecten vragen om een geleidelijke transitie.

Een goed voorbeeld van deze aanpak is hoe een inQdo-team experimenteerde met Projen in een nieuw serverless project. Dit project was klein genoeg om te overzien, maar complex genoeg om de mogelijkheden van Projen te testen. De eerste resultaten waren veelbelovend: waar het team normaal gesproken uren kwijt was aan het opzetten en configureren van een ontwikkelomgeving, stond er nu binnen enkele minuten een volledig geconfigureerd project.

Maar de echte waarde werd pas duidelijk toen er een update nodig was in de linting-regels. Een situatie die zich in de praktijk regelmatig voordoet: een team ontdekt een betere manier om code te structureren en wil deze standaard doorvoeren in alle projecten. Zonder Projen zou dit betekenen dat elk project handmatig moest worden aangepast. Met Projen was het een kwestie van de regel aanpassen in het centrale configuratiebestand en de wijziging automatisch laten doorvoeren.

Voordelen van Projen

Deze ervaring leidde tot een breder inzicht: Projen is niet alleen een tool voor projectconfiguratie, maar een manier om development standards te democratiseren. Door configuratie als code te behandelen, wordt het mogelijk om best practices direct in te bouwen in de projectstructuur.

Zo hoeven nieuwe teamleden niet meer door uitgebreide documentatie te spitten om te begrijpen hoe een project moet worden opgezet - de standaarden zijn ingebakken in de configuratie. Voorheen kostte het dagen om een nieuwe developer volledig productief te krijgen. Ze moesten zich verdiepen in verschillende configuratiebestanden, lokale ontwikkelomgevingen opzetten en projectspecifieke instellingen doorgronden. Nu is het een kwestie van één commando uitvoeren om een volledig geconfigureerde ontwikkelomgeving te hebben die voldoet aan alle teamstandaarden.

Een ander concreet voorbeeld is de integratie van security-tools. Door CDK-nag standaard op te nemen in de Projen configuratie, worden security checks een natuurlijk onderdeel van het development proces. De tool kan niet zomaar worden uitgeschakeld of omzeild, wat

zorgt voor consistente veiligheidsstandaarden over alle projecten heen. Maar ook binnen bestaande projecten zorgt Projen ervoor dat security eenvoudiger doorgevoerd kan worden. Neem bijvoorbeeld een recent project waarbij een kritieke security update moest worden doorgevoerd in alle repositories. Waar dit voorheen een volledige sprint in beslag zou nemen, werd het nu een gestroomlijnde operatie die binnen een dag was afgerond. Niet alleen was dit efficiënter, het elimineerde ook het risico dat een repository per ongeluk werd overgeslagen.

De praktijk leert ook dat teams verschillende niveaus van automatisering nodig hebben. Sommige projecten vereisen strikte controle over elke configuratie-instelling, terwijl andere meer flexibiliteit nodig hebben. Projen ondersteunt dit door het onderscheid tussen managed en unmanaged bestanden. Een goed voorbeeld hiervan is de behandeling van GitHub workflows: de basis-workflows voor testen en deployment worden automatisch gegenereerd en bijgewerkt, terwijl teams de vrijheid behouden om projectspecifieke workflows toe te voegen.

De automatisering van configuratiebeheer heeft een onverwacht maar positief effect op de efficiëntie van code reviews. In traditionele ontwikkelomgevingen zorgen inconsistente configuraties ervoor dat belangrijke codewijzigingen worden gemaskeerd door oppervlakkige formatteringsverschillen. Het is vergelijkbaar met het reviewen van een document waarin niet alleen de inhoud is aangepast, maar ook de opmaak willekeurig is gewijzigd - het wordt dan bijna onmogelijk om de werkelijke inhoudelijke veranderingen te identificeren. Projen elimineert dit probleem door consistente formatting af te dwingen.

Hierdoor worden functionele wijzigingen, zoals nieuwe features of security patches, direct zichtbaar in code reviews. Teams kunnen zich concentreren op wat echt belangrijk is: de logica en functionaliteit van de code. Deze verhoogde zichtbaarheid van relevante wijzigingen versnelt niet alleen het reviewproces, maar verbetert ook de kwaliteit van de reviews zelf.

Veelvoorkomende valkuilen en hoe je ze vermijdt

1. Vertrouwen op automatisering zonder begrip
Uitdaging: Teamleden vertrouwen volledig op Projen zonder te begrijpen hoe het werkt.
Oplossing: Combineer automatisering met training en documentatie, zodat teamleden begrijpen wat Projen doet en hoe ze het kunnen aanpassen.

2. Overweldigende complexiteit in het begin
Uitdaging: Te veel standaarden tegelijk implementeren kan verwarrend zijn.
Oplossing: Begin met een eenvoudige configuratie en voeg complexiteit geleidelijk toe op basis van feedback.

3. Onvoldoende communicatie binnen het team
Uitdaging: Niet iedereen in het team begrijpt de voordelen van Projen.
Oplossing: Bespreek regelmatig de impact van Projen en deel successen, zoals tijdwinst en verbeterde consistentie.

5

Een blik op de toekomst

Softwareontwikkeling is in constante evolutie. Teams worden geconfronteerd met toenemende complexiteit, snelle technologische veranderingen en de groeiende vraag naar gestandaardiseerde processen. Projectconfiguratie – ooit een simpele taak – is nu een strategische uitdaging geworden.

Trends die deze verandering aandrijven:

- **Schaalbaarheid:** Het aantal repositories en projecten binnen organisaties neemt toe, samen met de noodzaak om consistentie te waarborgen.
- **Automatisering:** Tools zoals CI/CD en AI ondersteunen ontwikkelaars, maar vereisen gestandaardiseerde configuraties om optimaal te presteren.
- **Diversiteit in teams:** Ontwikkelteams bestaan steeds vaker uit mensen met verschillende niveaus van technische expertise, wat de noodzaak voor gebruiksvriendelijke tools vergroot.

In dit landschap speelt Projen een sleutelrol door configuratie te behandelen als code en daarmee een fundamenteel probleem in softwareontwikkeling op te lossen.

Automatisering is een kernonderdeel van moderne softwareontwikkeling. Van het genereren van builds tot het uitvoeren van tests en deploys – ontwikkelaars vertrouwen op een breed scala aan geautomatiseerde tools. Projen sluit naadloos aan bij deze trend door configuraties te automatiseren. Met de opkomst van AI-tools zoals GitHub Copilot en andere code-assistenten, wordt het steeds belangrijker om duidelijke en consistente configuraties te hebben.

Een interessante ontwikkeling is de integratie van Projen met moderne development tools. Door de configuratie als code te behandelen, wordt het mogelijk om dezelfde tools te gebruiken voor configuratiebeheer als voor softwareontwikkeling. Dit betekent bijvoorbeeld dat teams kunnen profiteren van IntelliSense en type checking bij het aanpassen van projectconfiguraties - functionaliteit die voorheen alleen beschikbaar was voor applicatiecode.

De volgende stap in moderne softwareontwikkeling

De uitdagingen van projectconfiguratie zijn niet uniek. Elk ontwikkelteam dat groeit, herkent de worsteling met inconsistente configuraties, tijdrovende updates en moeizame onboarding van nieuwe teamleden. Wat wel bijzonder is, is de manier waarop Projen deze problemen aanpakt. Door projectconfiguratie te behandelen als code, transformeert het een traditioneel pijnpunt in een strategisch voordeel.

De praktijk wijst uit dat de impact verder gaat dan alleen efficiëntie. Teams ervaren een fundamentele verschuiving in hoe ze over projectconfiguratie denken. Het is niet langer een noodzakelijk kwaad, maar een integraal onderdeel van de development workflow. Deze mindshift leidt tot betere code, snellere ontwikkelcycli en meer focus op wat echt belangrijk is: het bouwen van waardevolle software.

6

Concrete stappen om te beginnen

Voor teams die de overstap naar Projen overwegen, is een gefaseerde aanpak essentieel. Begin klein, met één project waar het team de ruimte heeft om te experimenteren. Dit kan een nieuw project zijn of een bestaand project dat toe is aan een refresh. Het belangrijkste is dat het team de vrijheid heeft om te leren en aan te passen.

Besteed in deze fase vooral aandacht aan:

- Het definiëren van teamstandaarden die echt waardevol zijn
- Het identificeren van configuraties die het meeste baat hebben bij automatisering
- Het verzamelen van feedback van teamleden over wat wel en niet werkt

Zodra het eerste project succesvol draait met Projen, kun je beginnen met het uitbreiden naar andere projecten. De ervaring leert dat teams op dit punt vaak al zo overtuigd zijn van de voordelen, dat ze zelf vragen om meer projecten te migreren.

Een investering in de toekomst

De toekomst van softwareontwikkeling ligt in automatisering en standaardisatie. Tools als Projen spelen hierin een cruciale rol. Door projectconfiguratie te behandelen als een first-class citizen in het ontwikkelproces, leggen teams een solide fundament voor toekomstige groei en innovatie.
De integratie met moderne ontwikkeltools en de mogelijkheid om configuraties programmatisch te beheren, maakt Projen klaar voor de volgende golf van innovaties in softwareontwikkeling. Of het nu gaat om AI-assistenten, nieuwe security-eisen of nog onbekende technologische ontwikkelingen - een gestructureerde, code-first benadering van projectconfiguratie zorgt ervoor dat teams klaar zijn voor wat de toekomst brengt.

Start vandaag nog!
Wil je meer weten over hoe Projen jouw development processen kan verbeteren?

Neem dan contact op met onze experts:
info@inqdo.com
+31 85 2011161
inQdo.com

Start vandaag nog met het vereenvoudigen van je projectconfiguratie en ervaar zelf de kracht van Projen!

7

De belangrijkste lessen uit dit whitepaper

1. Waarom Projen?

Projen lost fundamentele problemen op die andere tools zoals Cookiecutter en GitHub Templates niet aanpakken:

- Het biedt continu beheer, in plaats van alleen initiële setups.
- Het elimineert handmatige updates door configuraties centraal te beheren.
- Het verhoogt de productiviteit en vermindert fouten door standaardisatie.

2. Hoe werkt Projen?

Projen maakt gebruik van een gecentraliseerd configuratiebestand (projenrc.ts) om consistentie te waarborgen. Dit bestand fungeert als de bron van waarheid voor elk project en automatiseert:

- Het aanmaken en bijwerken van bestanden zoals .gitignore, package.json, en CI/CD-workflows.
- Het integreren van (security-)tools zoals cdk-nag.
- Het implementeren van teamstandaarden voor linting en formattering.

3. Praktijkervaringen met Projen

Bij organisaties zoals inQdo heeft Projen bewezen waardevol te zijn door:

- Tijd te besparen bij het updaten van configuraties.
- De onboarding van nieuwe teamleden te versnellen.
- De samenwerking binnen teams te verbeteren door consistente standaarden te hanteren.

8

Aanbevolen bronnen voor verdere verdieping

Officiële documentatie van Projen:
projen.io
Hier vind je gedetailleerde handleidingen en voorbeelden.

GitHub-repository:
github.com/projen/projen
Bekijk de codebase, openstaande issues en de nieuwste releases.

Community-discussies:
Sluit je aan bij forums en communities om ervaringen uit te wisselen en best practices te ontdekken.

Over inQdo

inQdo is een AWS Advanced Consulting Partner gespecialiseerd in cloud solutions en digitale transformatie. Met een team van ervaren cloud developers helpen we organisaties bij het implementeren van innovatieve oplossingen die business value creëren.

info@inqdo.com
+31 85 2011161

---

inQdo
Coltbaan 1-19
3439 NG Nieuwegein