De type hiërarchie in Perspectives

|  |  |  |
| --- | --- | --- |
| Joop Ringelberg | 20-03-18 | Versie: 1 |

# Introductie

Elke context heeft een type. Dit type wordt in de representatie bijzonder behandeld, namelijk als veld van het Purescript Record waarmee we een Context representeren. We kunnen er echter over denken als een rol van een Context.

Een type wordt in Perspectives, in CRL, beschreven met een Context. Daarom heeft een type zelf ook een type (want het is een Context en die heeft altijd een type).

De hiërarchie die deze types vormen, benadrukken we door de buitenrol van een type te vullen met de buitenrol van zijn type. Zo representeert de buitenrol-telescoop van een type al zijn voorouders.

De buitenrol van een instantie vullen we óók met de buitenrol van zijn type. We kunnen namelijk geen onderscheid maken tussen contexten die wel en niet gebruikt worden als type.

## Type properties: ‘class level properties’

Properties van de buitenrol van een type kunnen we vanuit een instantie van zo’n type beschouwen als een ‘class level property’. We hebben echter geen classes in Purescript dus daarom noemen we dit type-properties. In de vorige paragraaf zagen we hoe de types een hiërarchie vormen. We kunnen type properties dus in de hele hiërarchie aflezen.

# Partiële instantiatie

Als we een context instantiëren, hoeven we niet alle properties of rollen die bij zijn type zijn gedefinieerd, aan te maken.

We beschouwen een instantie van een context waarvoor het type nog rollen (of properties) bevat die niet zijn aangemaakt, zelf ook als een type. We kunnen dit goed vergelijken met partiële applicatie bij functionele talen. Door de functie + toe te passen op het getal 1 verkrijgen we een functie van één parameter die bij elk argument 1 optelt.

Zo werkt het ook bij Contexten. Een type Rit dat twee rollen beschrijft, Chauffeur en Passagier, kunnen we partieel instantiëren met b.v. “Jansen” in de rol van Chauffeur. We verkrijgen zo een nieuw type Context, dat we “RittenMetJansen” zouden kunnen noemen.

Een instantie van RittenMetJansen kunnen we de rol passagier geven. Maar niet meer de rol chauffeur (of die rol nu functioneel is of niet).

En natuurlijk kunnen we een instantie waarvoor een property van de buitenrol nog niet is ingevuld, óók als type beschouwen.

# Rol- en propertyqueries

Merk op dat dit gevolgen heeft voor de uitvoering van een query. Allereerst is het zo dat we instanties van rollen niet alleen bij de instantie van de context zelf moeten zoeken, maar óók in zijn type-hiërarchie (maar altijd maar op één niveau!).

Ten tweede is het zo dat de gekwalificeerde naam van een rol niet noodzakelijk in de namespace van de context zelf hoeft te vallen. Denk maar aan het type “RittenMetJansen”. De rol met de lokale naam “Passagier” heeft als gekwalificeerde naam “Rit$Chauffeur” en niet “RitMetJansen$Chauffeur”.

Hetzelfde geldt voor properties van de buitenrol.

# Een constraint

Als een context als type is gebruikt, mogen we zijn rollen niet meer invullen. Hierdoor zouden rollen namelijk op verschillende niveaus in de type hiërarchie voorkomen. Het is onduidelijk wat daar de semantiek van zou moeten zijn.

Kortom, een context die als type wordt gebruikt, moet worden ‘bevroren’.