	Retour au Dossier Organisationnel
Sujets connexes	Proc#179 - Développement : règles et bonnes pratiques Validateurs DevExtreme (Validation Rules)

Mise en place des dx-validator

Les différents dx-validator

A l'intérieur d'une zone de saisie devetrem (<dx-text-box>, <dx-check-box>, <dx-date-box>...), il est possible de mettre en place des **validateurs de saisie**
il se présente sous la forme

```
  <dx-validator validationGroup="infoBanque">
       <dxi-validation-rule type="custom"
[validationCallback]="customValidator.isValidBicFromDx" message="Le BIC est
invalide"/>
  </dx-validator>
```

- **infoBanque** corresponds au nom d'un groupe de validateur (dans le cas général on peut utiliser le même pour tous ceux d'un composant simple
- **custom** corresponds au type de validation que l'on souhaite (cf plus loin)
    - ici custom = un validateur maison (il est déclaré dans [validationCallback]="customValidator.isValidBicFromDx")

Pour aller plus loin :

- Validateurs DevExtreme (Validation Rules)
- les validateurs "custum" **doivent** être déclaré dans la bibliothèque dans **custom-validation-rule.ts** et donc appelé via le service **customValidatorService**
- Cas particulier des dates

Test manuelle

```
    test_warning(){
        this.Notification.videWarning();
        if (this.personne.dateFinConservation) {
            const aujourdhui = new Date(); aujourdhui.setHours(0, 0, 0, 0);
            if (this.personne.dateFinConservation && new
Date(this.personne.dateFinConservation) < aujourdhui) {
                this.Notification.ajouteWarning('RGPD','La date de fin de
consentement de cette personne est dépassée')}
            else {
                const bientot = new Date(); aujourdhui.setHours(0, 0, 0, 0);
bientot.setMonth(bientot.getMonth()+6);
                if (this.personne.dateFinConservation && new
Date(this.personne.dateFinConservation) < bientot) {
```

```
                    this.Notification.ajouteWarning('RGPD','La date de fin
de consentement de cette personne va bientôt être dépassée')
                }
            }
        } else {this.Notification.ajouteWarning('RGPD',"La date de fin de
consentement de cette personne n'est pas initialisée")}
    }
```

```
private triggerValidation() {
  this.Notification.analyseStockValidation("infoIdentification","Information
d'identicateur");
  if ((this.SIREN=="")&&(this.RNA=="")) {
    this.Notification.ajouteErreur("infoIdentification","Veuillez saisir au
moins un des champs SIREN ou RNA")
  }
}
```

L'organisation des tests

Comme on l'a vu plus haut les validators sont regroupés en **validationGroup**
Dans le composant on va déclaré une série de procédures (pour une meilleur lecture on essaye de garder la nomenclature)

```
  private Notification= inject(NotificationService)
  ngAfterViewInit() {
    //votre code ici si besoin
    this.triggerValidation();
  }

  private triggerValidation() {
    this.Notification.analyseStockValidation("infoBanque","Information
bancaires");
  }
  handleInfoBanqueChanged() {
    // votre code si beoin
    this.triggerValidation();
    this.infoBancairesChange.emit(JSON.stringify(this.infoBanque))
  }
```

Le service "Notification"

Il publie plusieurs fonctions :

analyseStockValidation(validateGroup:string, nomBloc:string)	**validateGroup** est le nom du groupe de test (on peut en avoir plusieurs dans un même composant) **nomBloc** sert à l'affichage à indiquer à l'utilisateur ou se trouve les champs à corriger

From:
https://wiki-logeas.fr/certif/ - **dokuwiki-certif**

Permanent link:
https://wiki-logeas.fr/certif/doku.php?id=certif:procedure:develop:angular:validator&rev=1779356780

Last update: **2026/05/21 11:46**