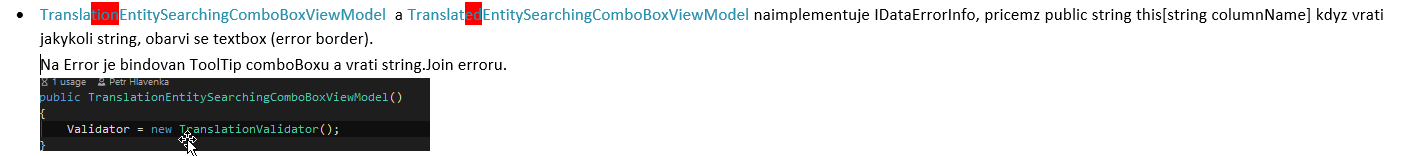
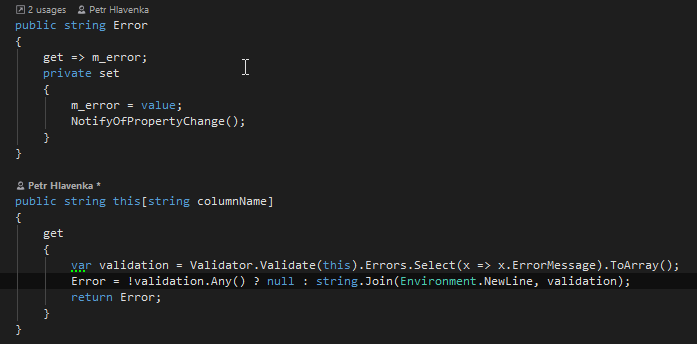
Validace FluentValidator

# 9.1.2020 Nejjednodussi validace pouzitim IDataErrorInfo (ExplanationSolution projekt IDataErrorInfoWithFluentValidation):

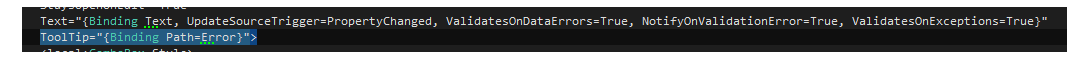
**Fluent validator jen plni stringovou propertu Error o notifikaci view se stara rozhrani IDataErrorInfo, ktere musi VM implementovat.**

Kodovadlo (b9f2c2daa96488c507b58db34796bbac37c86813)





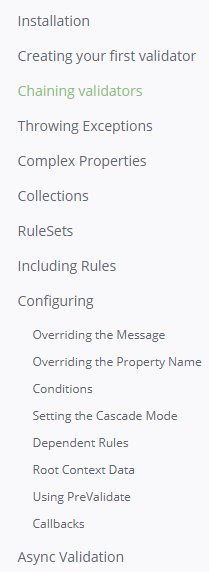
Null je dulezite, kvuli toolTipu. Pokud bych totiz vracel string.Empty, tak by se mi zobrazoval toolTip ale byl by prazdny. Takhle se mi nezobrazi vubec pokud nemam error.





# Prehled

<https://fluentvalidation.net/>



# Pouziti validace <motivlet> pro validaci stringu v jinem view modelu

<https://fluentvalidation.net/start#complex-properties>

# Validace ( prekopirovano z WordPadu z Vesmiru + neco navic)

‎17 ‎August, ‎2017

* **VALIDACE :**
* [**https://github.com/JeremySkinner/FluentValidation/wiki**](https://github.com/JeremySkinner/FluentValidation/wiki)

using FluentValidation;

public class CustomerValidator : AbstractValidator<Customer> **dedime od Abstraktni tridy**

{

public CustomerValidator() **// Vytvorime nejake pravidla validace**

{

RuleFor(customer => customer.Surname).NotEmpty();

RuleFor(customer => customer.Forename).NotEmpty().WithMessage("Please specify a first name");

RuleFor(customer => customer.Discount).NotEqual(0).When(customer => customer.HasDiscount);

RuleFor(customer => customer.Address).Length(20, 250);

RuleFor(customer => customer.Postcode).Must(BeAValidPostcode).WithMessage("Please specify a valid postcode");

}

private bool BeAValidPostcode(string postcode)

{

// custom postcode validating logic goes here

}

}

Customer customer = new Customer(); **// Nova instance customera**

CustomerValidator validator = new CustomerValidator(); **// Nova instance validatoru**

ValidationResult **results** = validator.Validate(customer); **// Pomoci metody rodice validujeme customera**

bool validationSucceeded = **results**.IsValid; **// Boolean true nebo false podle toho jestli validace prosla**

IList<ValidationFailure> failures = **results**.Errors; **// Seznam chyb ke kterym doslo pri validaci, podle nasich pravidel.**

============================================================================================================================================================================

Vesmir

* Trida ktera chce pouzivat validaci (tady GalaxyDialogViewModel ) musi implementovat rozhrani IDataErrorInfo.
* Toto rozhrani ma dve property :

1. string this[string columnName] { get; } to je pole stringovych objektu IDataErrorInfo
2. string Error { get; }

* Implementaci rozhrani budou tyto dve property i v nasi tride .
* Do tridy si pridame vnitrni tridu CustomValidator : AbstractValidator<GalaxyDialogViewModel>
* Kde v konstruktoru budou pravidla ktere chceme zkoumat.
* A budeme tu mit propertu: protected IValidator Validator { get; } = new CustomValidator(); na ktere budeme validovat.
* Pozdeji jsem to predelal tak , aby se dedilo od tridy ErrorBase . Ta dedi od ViewModelBase kvuli rozhrani INotify , takze tridy ktere budou dedit od ErrorBase uz budou mit vyreseno INotify, IDataErrorInfo a budou uz mit validaci v predkovi.
* Na samotne tride ktera bude chtit neco validovat , pak zustene jen vytvorit vnitrni tridu CustomValidator a v jejim konstruktoru definovat co se bude testovat. :

RuleFor(d => d.Jmeno).NotEmpty().WithMessage("Jmeno musi byt vyplneno.");

* V konstruktoru jen zavolam metodu na predkovi Validate a jako parametr ji predhodim instanci sveho customValidatoru :

Validate(new CustomValidator());

* V xaml musime na komponente ktera se bude validovat zadefinovat:

1. UpdateSourceTrigger=PropertyChanged => ViewModel potrebuje vedet o tom ze se komponenta zmenila , jinak by nemohl vedet o chybe.
2. ValidatesOnDataErrors=True =>
3. NotifyOnValidationError=True =>
4. ToolTip="{Binding Path=Error}" => Vypis napovedy pro uzivatele definovany v customValidatoru

* Take je nutne notifikovat view o zmenach ve viewModelu . K tomu nam slouzi INotify s tim ze neinformujeme o zmene vlastnosti ale o zmene booleanu **IsValid** a stringu **Error**.:

OnPropertyChanged(nameof(Error));

OnPropertyChanged(nameof(IsValid));

* Povoleni tlacitek je nastaveno v RelayCommandu na druhem parametru, da se to udelat i v xaml a to nasledovne: (nefunguje bzzz)

CommandParameter="{Binding Path=IsValid}"

**VE SKENOVADLE JE POUZITY NA EDITPUBLICATIONVIEWMODELU , DEDI OD ERRORBASE**

**Ukoly:**

* ~~Ramecek v okne kde se budou vypisovat chyby~~
* Dovalidovat okna
* ~~Zablokovat commandy dokud nebude okno validni.~~

======================================

**Otazky :**

* ~~Presunuti validace na tridu predka mi zacalo misto errorStringu davat jen pole~~ **~~System.char[]~~**
* Co presne znamena to this public string this[string columnName]
* Proc mi nejde vymenit v RelayCommandu (GalaxyDialogViewModel) tato metoda za lambda expr. ? ()=> true Jak napsat predikat ? Proc nemuzu odebrat ten parametr ?

private bool CanSaveCommand( object parameter)

{

return IsValid;

}

* Command="{Binding SaveGalaxyCommand}" CommandParameter="{Binding Path=IsValid}"
* Mam spravne hierarchii slozek ? Proc Mitroz tvoril assembly v podobe dalsich projektu v Solution?
* 2. ValidatesOnDataErrors=True =>
* 3. NotifyOnValidationError=True =>
* nejaky toolkit na dva monitory
* projit jednotlive okna ve visual studiu + klavesove zkratky
* Vybrani vice radku v tabulce , eventy na mysi , pretahovani, kopirovani , vkladani.. , kontextove menu ,
* eventy v MVVM naprikla lostFocus
* XPath

Pokud chceme pouzit RelayCommand musi to byt instance ICommand.

* Sealed => neprepisovatelne
* Pokud nechci vypisovat vsude v setrech vlastnosti ze se zmenil error : OnPropertyChanged( nameOf(Error)); a nap dalsi veci ( IsValid) , muzu si udelat OnPropertyChangedMetodu v predkovi virtualni . Dedime z tridy ViewModelBase , tam je implementovano INotify.

Puvodni metoda OnPropertyChanged uz nevyvolava event , vola jeno dve nove metody : **virtual OnPropertyChangedInternal** a **OnPropertyChangedEx .**

Virtualni metoda OnPropertyChangedInternal je urcena k prepisovani v potomkovi - override , v rodicovske tride je prazdna.

Protected metod OnPropertyChanged**Ex** prebira cinnost puvodni metody : vyhazovani eventu PropertyChanged.

Ve **tride potomka:** ( tady trida ErrorBase ) overridneme virtualni metodu predka OnPropertyChanged a pritom tim rekneme co se v ni ma navic stat. V tomto pripade trida ErrorBase resi validaci takze potrebujeme aby jeji potomci ( dialogove okna ) , zavolanim metody OnPropertyChanged( nameOf( nazevVlastnosti)); navic provedli informovani View o zmenach na stringu Error a na booleanu teto tridy IsValid. :

OnPropertyChangedEx(nameof(Error));

OnPropertyChangedEx(nameof(IsValid));

To pro nas znamena , ze uz tyto veci nemusime vypisovat pri kazde vlastnosti kterou chceme notifikovat.

===================

‎

# Validaci jde zpusobit i bez validatoru (Zmenovadlo) :

2/‎7/‎2018  **Zmenovadlo:**

xmlns:errors="clr-namespace:MIR.Media.Changing2.Gui.Errors"

<DatePicker

Name="DatePicker"

Text="{Binding SelectedDate}" >

<DatePicker.SelectedDate>

<Binding Path="SelectedDate" NotifyOnValidationError="True">

<Binding.ValidationRules>

<errors:DatePickerDateValidationRule/>

</Binding.ValidationRules>

</Binding>

</DatePicker.SelectedDate>

</DatePicker>

**a k tomu validationRule:**

using System;

using System.Globalization;

using System.Windows.Controls;

namespace MIR.Media.Changing2.Gui.Errors

{

public class DatePickerDateValidationRule : ValidationRule

{

public override ValidationResult Validate(object value, CultureInfo cultureInfo)

{

if (value != null)

{

var date = (DateTime) value;

if (date > new DateTime(2000, 1, 1))

{

return ValidationResult.ValidResult;

}

return new ValidationResult(false, "Blbost");

}

return ValidationResult.ValidResult;

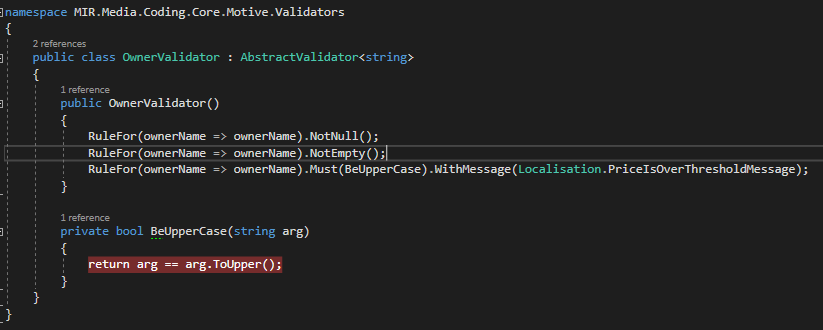
}

}

}

# Kodovadlo uz celkem dobre vysvetleno :

* Musim nareferencovat FluentValidator
* Vytvorim tridu ktera podedi AbstractValidator



Validace se pak da pouzit v kodu takto:

if (CreateNewOwner)

{

var newName = ByNameOwnerSearchingComboBox.Text.Trim();

var validator = new OwnerNameValidator();

var result =: validator.Validate(newName).IsValid;

if (!IsCzLocation && result == false)

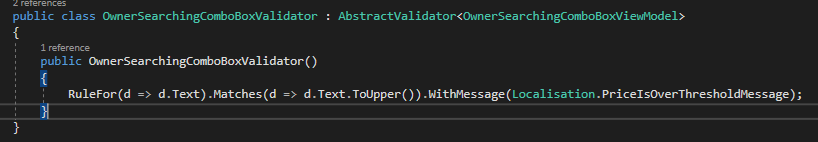
{

Message = Localisation.OnlyUpperCaseLettersAreAllowed;

return;

}

* Pokud chci zobrazovat error info ve view musim to udelat takto : To horni plati toto je jina verze jako argument si muzu dat objekt comboboxu



* V xamlu si v komponente kterou chci validovat dame

<Grid>

<SearchingComboBox1:ComboBox

Text="{Binding Text, UpdateSourceTrigger=PropertyChanged, ValidatesOnDataErrors=True, NotifyOnValidationError=True}" ToolTip="{Binding Path=Error}">

</SearchingComboBox1:ComboBox>

</Grid>

* A v modelu tohoto comboboxu ( cela trida) nebo v modelu ktery patri k view implementujeme IdataErrorInfo:

namespace MIR.Media.Coding.Core.Motive.SearchingComboBox

{

public class OwnerSearchingComboBoxViewModel : SearchingComboBoxViewModelBase<Owner> , IDataErrorInfo

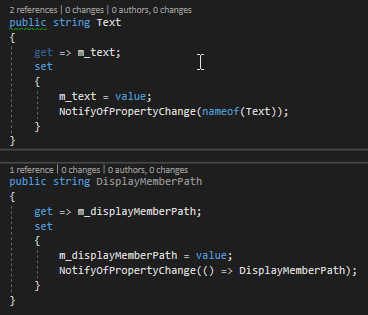
{

public OwnerSearchingComboBoxViewModel()

{



}





public string Error

{

get

{

if (Validator == null)

return string.Empty;

var validation = Validator.Validate(this).Errors.Select(x => x.ErrorMessage).ToArray();

if (!validation.Any())

return string.Empty;

return string.Join(Environment.NewLine, validation);

}

}

Sem se vlozi objekt ktery se bude validovat. Klidne to muze byt nejaka properta na tride, ktera implementuje IDataErrorInfo. Genericky argument validatoru pak musi mit typ jako tato properta.

public string this[string columnName]

{

get

{

if (Validator == null)

return string.Empty;

var validation = Validator.Validate(this).Errors.FirstOrDefault(e => e.PropertyName == columnName);

NotifyOfPropertyChange(nameof(Error)); // aby se zobrazil tool tip ve view

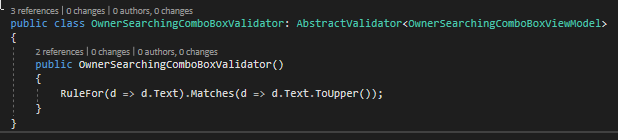
return validation == null ? string.Empty : validation.ErrorMessage;

}

}

}

* Tot vse tady se zadne Validator.Validate nevola zvyraznovani stejne funguje.
* Je ovsem potreba mit napsany validator, zde je to OwnerSearchingComboBoxValidator:



* Dalsi pouziti v Kodovadle:

TranslationEntitySearchingComboBoxViewModel

UserStory: <https://tfs.mediaresearch.cz/AdIntel/AdIntel/_workitems/edit/51494>

Reseni: 9074c537f1738ef8a748b7dc2ca3cead66897c19

# Rules

RuleFor(m => m.Text).NotEmpty().WithMessage(Localisation.NotAllParametersProvided); **// string neni prazdny**

RuleFor(m => m.Text).Must(n => n.Length < StringPropertyCharLimit).Unless(m => m.Text == null).WithMessage(string.Format(Localisation.PropertyCharValueOverlength, string.Empty, StringPropertyCharLimit)); **// string nepresahuje stanoveny pocet znaku**

RuleFor(m => m.Text).Must(n => n != null && !Regex.IsMatch(n, @"\p{IsCyrillic}+")).Unless(m => m.Text == null).WithMessage(string.Format(Localisation.TranslationCouldNotContainCyrilic, Localisation.Value)); **// string neobsahuje cyrilici (BG)**

RuleFor(m => m.Text).Must(d => !d.StartsWith(" ")).Unless(m => m.Text == null).WithMessage(Localisation.TranslationNotContainsSpace); **// string nezacina prazdnym znakem**

RuleFor(m => m.Text).Must(d => !d.EndsWith(" ")).Unless(m => m.Text == null).WithMessage(Localisation.TranslationNotContainsSpace); **// string nekonci prazdnym znakem**

RuleFor(m => m.Text).Must(d => !d.Trim().Contains(" ")).Unless(m => m.Text == null).WithMessage(Localisation.TranslationNotContainsInnerSpace); **// string nema uprostred prazdny znak s vyjimkou, kdy text je null**

RuleFor(m => m.Text).Must(d => d).Must(BeUpperCase).WithMessage(…) **// string musi byt velkymi pismeny**

RuleFor(customer => customer.Discount).NotEqual(0).When(customer => customer.HasDiscount);

RuleFor(customer => customer.Address).Length(20, 250);

RuleFor(customer => customer.Postcode).Must(BeAValidPostcode).WithMessage("Please specify a valid postcode");

RuleFor(x => x.student\_id).Matches("^\d{7}$")....

# [How to validate only 7 digit number?](https://stackoverflow.com/questions/12907242/how-to-validate-only-7-digit-number)

<https://stackoverflow.com/questions/12907242/how-to-validate-only-7-digit-number>

Since you're using FluentValidation, you want to use the .Matches validator to perform a regular expression match.

RuleFor(x => x.student\_id).Matches("^\d{7}$")....

.