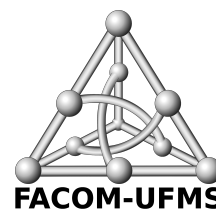




Universidade Federal de Mato Grosso do Sul (UFMS)
Faculdade de Computação (FACOM)



Documentação Técnica das

Regras JSONForms

(Rule)

Campo Grande/MS
2024

Introdução

Este documento visa fornecer uma documentação técnica clara e concisa sobre as regras e funcionalidades avançadas do JSONForms, com foco em exemplos práticos e cenários de teste. A documentação segue as diretrizes de Technical Writing do Google, buscando clareza, precisão e facilidade de compreensão para desenvolvedores e usuários.

O JSONForms é uma ferramenta poderosa para a criação de formulários dinâmicos a partir de JSON Schemas. Compreender suas regras é essencial para construir interfaces de usuário robustas e eficientes. Esta documentação abordará as seguintes regras e conceitos:

- Visibilidade Condicional (**rule** com **effect: HIDE/SHOW**)
- Múltiplas Condições (**allOf** , **anyOf**)
- Regras Avançadas - **Tenho que ver essa ainda!**

Cada seção apresentará a regra, sua aplicação em um JSON Schema e UI Schema, e um cenário de teste detalhado para demonstrar seu comportamento. O objetivo é fornecer um recurso abrangente que complemente a documentação oficial do JSONForms, preenchendo lacunas e oferecendo exemplos práticos para facilitar o aprendizado e a implementação.

1. Visibilidade Condicional Simples (**rule** com **effect: HIDE/SHOW**)

A regra de visibilidade condicional no JSONForms permite controlar a exibição de elementos do formulário (campos, grupos, layouts) com base em condições definidas no JSON Schema. Utilizando a propriedade **rule** dentro do **uischema** , é possível especificar um **effect** (como **HIDE** para ocultar ou **SHOW** para exibir) e uma **condition** que, quando verdadeira, aplica o efeito. Essa funcionalidade é crucial para criar formulários dinâmicos e responsivos, que se adaptam às entradas do usuário, melhorando a experiência e a relevância dos campos apresentados.

Cenário de Teste: Solução - Visibilidade Condicional Simples

Este cenário demonstra como um campo de telefone pode ser exibido ou ocultado dinamicamente com base na resposta do usuário a uma pergunta simples: "Deseja receber ligações?". O objetivo é garantir que o campo de telefone só seja visível e, portanto, preenchível, quando o usuário explicitamente indicar que deseja receber ligações.

JSON Schema

```
{
  "type": "object",
  "properties": {
    "nome": {
      "type": "string",
      "title": "Nome Completo",
      "minLength": 3
    },
    "receberLigacoes": {
      "type": "string",
      "title": "Deseja receber ligacoes?",
      "enum": [
        "Sim",
        "Nao"
      ],
      "default": "Nao"
    },
    "telefone": {
      "type": "string",
      "title": "Telefone",
      "pattern": "^\\([0-9]{2}\\)\\[0-9]{4,5}-[0-9]{4}$"
    }
  },
  "required": [
    "nome",
    "receberLigacoes"
  ]
}
```

UI Schema

```
{
  "type": "VerticalLayout",
  "elements": [
    {
      "type": "Control",
      "scope": "#/properties/nome",
      "label": "Nome Completo"
    },
    {
      "type": "Control",
      "scope": "#/properties/receberLigacoes",
      "label": "Deseja receber ligacoes?"
    },
    {
      "type": "Control",
      "scope": "#/properties/telefone",
      "label": "Telefone",
      "rule": {
        "effect": "HIDE",
        "condition": {
          "scope":
            "#/properties/receberLigacoes",
          "schema": {
            "not": {
              "const": "Sim"
            }
          }
        }
      }
    }
  ]
}
```

Explicação da Solução

No `uischema`, a propriedade `rule` é aplicada ao controle do campo `telefone`. Esta regra define que o campo deve ser `HIDE` (ocultado) quando a `condition` for verdadeira. A condição, por sua vez, verifica o `scope #/properties/receberLigacoes` e utiliza `schema: { "not": { "const": "Sim" } }`. Isso significa que o campo `telefone` será ocultado se o valor de `receberLigacoes` não for "Sim". Consequentemente, o campo `telefone` só será exibido quando `receberLigacoes` for exatamente "Sim".

Passos para Teste

- 1. Inicie o formulário:** Observe que o campo "Telefone" não está visível inicialmente, pois o valor padrão de "Deseja receber ligações?" é "Não".
- 2. Altere a opção:** Selecione "Sim" para "Desejar receber ligações?". O campo "Telefone" deve aparecer imediatamente.
- 3. Preencha o telefone:** Insira um número de telefone válido no formato (XX) XXXX-XXXX ou (XX) XXXXX-XXXX.
- 4. Altere de volta:** Selecione "Não" para "Desejar receber ligações?". O campo "Telefone" deve desaparecer.
- 5. Verifique a validação:** Tente enviar o formulário com o campo "Telefone" visível, mas vazio ou com formato inválido, e observe as mensagens de erro.

Este teste valida a capacidade do JSONForms de controlar a visibilidade de campos de forma condicional, garantindo que apenas informações relevantes sejam solicitadas ao usuário em um determinado momento.

2. Múltiplas Condições (`allOf`, `anyOf`)

No JSON Schema, é possível combinar múltiplas condições lógicas para controlar a visibilidade ou o comportamento de elementos do formulário. As palavras-chave `allOf` e `anyOf` são usadas para expressar essas combinações:

- **`allOf`** : Significa que todas as subcondições especificadas devem ser verdadeiras para que a condição principal seja satisfeita. É o equivalente lógico de um operador AND.

- **anyOf** : Significa que pelo menos uma das sub-condições especificadas deve ser verdadeira para que a condição principal seja satisfeita. É o equivalente lógico de um operador OR.

Essas combinações são poderosas para criar lógicas de formulário complexas, onde a exibição de um campo depende de várias seleções ou estados de outros campos.

Cenário de Teste: Teste 3 - Múltiplas Condições

Este teste explora como um campo pode ser exibido apenas quando múltiplas condições são satisfeitas simultaneamente. O objetivo é mostrar o campo "Detalhes do Evento" somente se o usuário desejar participar de eventos E o tipo de evento selecionado for "Presencial".

JSON Schema

```
{
  "type": "object",
  "properties": {
    "nome": {
      "type": "string",
      "title": "Nome Completo",
      "minLength": 3
    },
    "participarEventos": {
      "type": "string",
      "title": "Deseja participar de eventos?",
      "enum": [
        "Sim",
        "Nao"
      ],
      "default": "Nao"
    },
    "tipoEvento": {
      "type": "string",
      "title": "Tipo de Evento",
      "enum": [
        "Presencial",
        "Online",
        "Hibrido"
      ],
      "default": "Online"
    }
  }
}
```

UI Schema

```
{
  "type": "VerticalLayout",
  "elements": [
    {
      "type": "Control",
      "scope": "#/properties/nome",
      "label": "Nome Completo"
    },
    {
      "type": "Control",
      "scope": "#/properties/participarEventos",
      "label": "Deseja participar de eventos?"
    },
    {
      "type": "Control",
      "scope": "#/properties/tipoEvento",
      "label": "Tipo de Evento"
    },
    {
      "type": "Control",
      "scope": "#/properties/detalhesEvento",
      "label": "Detalhes do Evento",
      "rule": {
        "effect": "SHOW",
        "condition": {
```

```

"detalhesEvento": {
  "type": "string",
  "title": "Detalhes do Evento",
  "description": "Informacoes sobre
local, horario e programacao",
  "minLength": 10
},
"endereco": {
  "type": "string",
  "title": "Endereco para Eventos
Presenciais",
  "minLength": 10
},
"required": [
  "nome",
  "participarEventos",
  "tipoEvento"
]
}

```

```

"schema": {
  "allOf": [
    {
      "properties": {
        "participarEventos": {
          "const": "Sim" }
      },
      "required":
["participarEventos"]
    },
    {
      "properties": {
        "tipoEvento": { "const":
"Presencial" }
      },
      "required": ["tipoEvento"]
    }
  ]
},
{
  "type": "Control",
  "scope": "#/properties/endereco",
  "label": "Endereco para Eventos
Presenciais"
}
]
}

```

Explicação da Solução

Para o campo *detalhesEvento*, a *rule* no *uischema* utiliza *effect: SHOW* combinado com *allOf* na *condition*. Isso significa que o campo só será visível se o valor de *participarEventos* for "Sim" E o valor de *tipoEvento* for "Presencial". SSe qualquer uma dessas condições não for atendida, o campo permanecerá oculto.

O campo *endereco* utiliza uma regra mais simples, sendo visível apenas quando *tipoEvento* for "Presencial".

Passos para Teste

1. **Inicie o formulário:** Observe que o campo "Detalhes do Evento" e

"Endereço para Eventos Presenciais" não estão visíveis.

2. **Selecione "Sim" para "Deseja participar de eventos?":** O campo "Detalhes do Evento" ainda não deve aparecer, pois a segunda condição (`tipoEvento = "Presencial"`) não foi atendida.
3. **Mantenha "Sim" e selecione "Presencial" para "Tipo de Evento":**
Agora, o campo "Detalhes do Evento" deve aparecer. O campo "Endereço para Eventos Presenciais" também deve aparecer.
4. **Altere "Tipo de Evento" para "Online":** O campo "Detalhes do Evento" e "Endereço para Eventos Presenciais" devem desaparecer.
5. **Altere "Deseja participar de eventos?" para "Não":** O campo "Detalhes do Evento" deve desaparecer, mesmo que "Tipo de Evento" seja "Presencial".

Este teste valida a capacidade do JSONForms de lidar com múltiplas condições para controlar a visibilidade de campos, o que é fundamental para formulários complexos e lógicas de negócio.

... Continuar com as regras avançadas!