ALPHINAT®

SmartGuide® 7.1

GUIDE DE DÉVELOPPEMENT .NET

Alphinat SmartGuide® — Guide du développeur .Net
Février 2018

Les informations contenues dans ce guide sont fournies à titre purement informatif ; elles sont susceptibles d'être modifiées sans préavis et ne doivent pas être interprétées comme étant un engagement de la part d'Alphinat Inc. Alphinat Inc. n'accepte aucune responsabilité quant aux erreurs ou inexactitudes pouvant être contenues dans le présent guide. Veuillez noter que les illustrations et images existantes que vous souhaiterez éventuellement inclure dans votre projet sont susceptibles d'être protégées par les lois sur les droits d'auteur. L'inclusion non autorisée de tels éléments dans vos nouveaux travaux peut constituer une violation des droits du propriétaire. Veuillez vous assurer de détenir toute autorisation nécessaire auprès du détenteur des droits. Toute référence à des noms et à des logos de société dans le matériel ou les formulaires d'exemple inclus dans le présent logiciel n'est faite qu'à titre de démonstration et ne vise aucun organisme réel.

Alphinat Inc., 2000 Peel, Suite 680, Montreal (Qc), H3A 2W5, Canada.

Table des matières

Introduction

1

À propos de SmartGuide

Que vous cherchiez à automatiser vos processus à base de formulaires, à créer des applications composites (mashups) ou à mécaniser des tâches répétitives, la réussite passe inévitablement par l'adoption du nouveau système par vos utilisateurs cibles. Et la meilleure façon de s'en assurer est de bâtir des applications qui soient faciles à utiliser et qui répondent parfaitement à l'ensemble de leurs besoins. Alphinat SmartGuide permet de créer des applications web qui les guident avec une précision de 100% dans leur recherche d'information et l'accomplissement de tâches. Cela permet de transposer dans un environnement web la convivialité et l'efficacité des interactions en personne.

Étendre SmartGuide

Alphinat SmartGuide offre un système d'extension qui permet d'ajouter des fonctionnalités et applications aux Smartlets créés avec SmartGuide Designer. Cela peut être le simple ajout d'une fonction permettant de changer la casse d'un texte, ou alors l'ajout de grosses fonctionnalités comme par exemple la sauvegarde des données saisies dans une base de données. Une fonction d'extension peut aussi permettre l'envoi par courrier électronique d'un PDF dynamiquement généré. Ou encore, une fonction d'extension peut effectuer une série de manipulations avancées sur des champs. Un simple fichier de configuration permet ensuite à des non-techniciens d'utiliser ces nouvelles fonctions au sein de SmartGuide Designer. Ces fonctions peuvent alors être utilisées afin d'initialiser un Smartlet, lors de la navigation d'une page à une autre, pour valider un champ ou être appelées lorsque l'utilisateur-final clique sur un bouton par exemple.

Contenu du guide

Ce guide comporte une description détaillée des différentes étapes de création d'une fonction d'extension en .Net. Vous y retrouverez également deux exemples de fonctions ainsi qu'une référence complète des fonctions exposées par l'API SmartGuide.

À qui s'adresse ce guide

Ce guide s'adresse aux développeurs .Net qui créeront les fonctions d'extension qui seront utilisées au sein de SmartGuide Designer. Le guide assume une connaissance au moins sommaire du langage de programmation C#. Au besoin, le lecteur et invité à consulter les liens suivants avant de poursuivre.

Références

- http://msdn.microsoft.com/library/default.aspx
- http://msdn.microsoft.com/en-us/vstudio/hh388573
- http://msdn.microsoft.com/en-us/vstudio/hh341490

Mise en route 2

Configuration requise

Le développement de fonctions d'extension .Net nécessite la version 2.0 ou plus du Framework.

Il est possible de coder les fonctions d'extension simples avec un éditeur de base tel que Crimson Editor ou Notepad+. Pour les fonctions plus complexes, un environnement de développement intégré (EDI) tel que Visual Studio est recommandé.

Important : la version du Framework .Net utilisée pour développer les fonctions d'extension ne doit pas être plus récente que celle sur laquelle roule SmartGuide Server.

Expertise requise

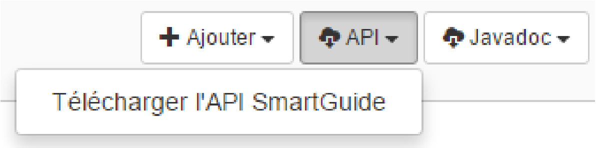
Le niveau d'expertise requis variera en fonction de la complexité de la fonction d'extension à développer. L'utilisation de l'API SmartGuide ne requiert pas d'expertise additionnelle particulière.

Installation de l'API SmartGuide

Pour développer des fonctions d'extension qui accèdent aux données d'un Smartlet ou au contexte dans lequel il est déployé, l'API de SmartGuide est nécessaire.

➤ **Pour téléchargez l'API:**

1. Accédez à **SmartGuide Designer**.
2. Cliquez sur l'onglet **connexions** dans la barre de navigation principale.
3. Cliquez sur le bouton API et sélectionnez l'option **Télécharger l'API SmartGuide**:



4. Sauvegardez le fichier sur votre disque dur sous le répertoire de votre choix.

Création d'une fonction

3

Introduction

Ce chapitre vous guidera pas à pas à travers le processus de création d'une fonction d'extension.

Développement des classes

Si vous développez une fonction d'extension qui fait appel à l'API SmartGuide, assurez-vous d'avoir une variable de type **IServiceContext** à type de paramètre d'entrée tel que démontré dans l'exemple suivant:

```
using System;
using System.Collections.Generic;
using System.Text;
using com.alphinat.sg5;

namespace Mybusiness
public class Utils {
  public static String toUppercase(IServiceContext context,
      String fieldName) {
    ISmartlet smartlet = context.getSmartlet();
    ISmartletField field = smartlet.findFieldByName(fieldName);
    if (field != null)
        return field.getString().ToUpper();
    return "";
  }
}
}
```

Veuillez consulter la section Référence API pour une description détaillée des fonctions offertes par l'API SmartGuide.

Création d'un fichier descriptif XML

La fonction d'extension doit contenir un fichier XML avec le nom apn-extension-function.xml, inséré en tant que ressource embarquée dans le dll. Ceci permet d'exposer ses méthodes et paramètres dans SmartGuide Designer. Le fichier XML doit avoir la structure suivante:

```
<classes>
  <class>
                <name>MyBusiness.Utils</name>
```

```
    <methods>
      <method>
        <name> toUppercase </name>
        <parameters>
          <parameter>
            <name>fieldName</name>
          </parameter>
        </parameters>
        <return>
          <name>value</name>
        </return>
      </method>
    </methods>
  </class>
</classes>
```

Veuillez noter que le nom de la classe, ainsi que le nom des paramètres et des valeurs de retour d'une fonction seront présentés aux utilisateurs de SmartGuide Designer lorsqu'ils feront appel à cette fonction. Il est donc recommandé d'utiliser des noms descriptifs. Noter également qu'il n'est pas nécessaire de déclarer le paramètre *context* de type *IServiceContext*.

Une fois créé, le fichier *apn-extension-function.xml* doit être marqué comme "Embedded resources".

Compiler la fonction d'extension

Simplement compiler le projet. Ceci devrait générer un dll nommé selon votre projet de fonction d'extension.

Si votre fonction d'extension utilise des librairies autres que celles de SmartGuide, la méthode suggérée est de placer ces librairies sous le répertoire **bin** de SmartGuide Server. Une autre option serait de mettre ces dépendances dans le Global Assembly Cache (GAC).

Faire appel à la fonction d'extension

Grâce au fichier descriptif XML, votre nouvelle fonction d'extension peut maintenant être utilisée au sein de SmartGuide Designer par des ressources non-techniques. Il leur suffira en effet de télécharger le fichier dll et de faire l'association des paramètres et valeurs de retour avec les champs d'un Smartlet.

Veuillez vous référer au guide de l'utilisateur SmartGuide Designer pour des instructions détaillées.

Exemples 4

Fonction d'extension "SetPageFieldsReadOnly"

La fonction d'extension suivante permet de marquer tout les champs de la page courante en mode de lecture seule.

Téléchargement de l'API SmartGuide

Assurez-vous d'avoir téléchargé l'API SmartGuide conformément aux instructions de la section Installation de l'API SmartGuide.

Développement des classes

- Créez la classe **Utils.cs**:

```
using System;
using System.Collections.Generic;
using System.Text;
using com.alphinat.sg5;

namespace MyBusiness
  public class Utils {
    public static void SetPageFieldsReadOnly(IServiceContext context) {
      ISmartletPage currentPage = context.getSmartlet().getCurrentPage();

      ISmartletField[] fields = currentPage.findAllFields();
      foreach(ISmartletField f in fields) {
        if (! (f.getTypeConst() == Constants.ElementType.GROUP ||
        f.getTypeConst() == Constants.ElementType.REPEAT ||
         f.getTypeConst() == Constants.ElementType.BUTTON ||
          f.getTypeConst() == Constants.ElementType.STATIC_IMG ||
           f.getTypeConst() == Constants.ElementType.STATIC_TEXT)) {
                f.setReadonly(true);
        }
      }
    }
  }
}
```

Création du fichier descriptif XML

Le fichier *apn-extension-function.xml* représentant la classe *Utils* expose la méthode *setPageFieldsReadOnly* dans SmartGuide Designer.

- Ajoutez le fichier **apn-extension-function.xml** à votre projet

```
<?xml version="1.0" encoding="UTF-8"?>
<classes>
  <class>
    <name>MyBusiness.Utils</name>
    <methods>
      <method>
      <name>SetPageFieldsReadOnly</name>
      <parameters>
      </parameters>
      </method>
    </methods>
  </class>
</classes>
```

Compilation de la fonction d'extension

Compilez votre projet pour obtenir une dll pour votre librairie de classes.

Reste ensuite à faire appel à la fonction d'extension à partir de SmartGuide Designer. Veuillez vous référer à la section 11, *Appels à des services* du guide de l'utilisateur SmartGuide Designer pour des instructions détaillées.

Fonction d'extension "SaveToFile"

La fonction d'extension suivante peut être utilisée afin de sauvegarder les données saisies par l'utilisateur dans un fichier texte dont le nom, incluant le répertoire, est passé en paramètre.

Téléchargement de l'API SmartGuide

Assurez-vous d'avoir téléchargé l'API SmartGuide conformément aux instructions de la section Installation de l'API SmartGuide.

Développement des classes

- Créez la classe **Utils.cs**:

```
using System;
using System.Collections.Generic;
using System.Collections;
using System.Text;
using com.alphinat.sg5;

using System.IO;
using com.alphinat.sg5.widget.group;
using com.alphinat.sg5.widget.repeat;

namespace MyBusiness
  public class Utils {
    public static void SaveToFile(IServiceContext context,
                string fileName) {
      // Map that will contain all the fields
      Hashtable mFields = new Hashtable();
      // Get Smartlet
      ISmartlet smartlet = context.getSmartlet();
      // Get all fields through finder
      SmartletFieldFindAll finder = new SmartletFieldFindAll();
      smartlet.accept(finder);

      ArrayList fields = finder.getTarget();
      if (fields == null)
        return;

      for(int i=0;i<fields.Count;i++) {
        ISmartletField f = (ISmartletField)fields[i];
        if (!f.isAvailable())
          continue;
        if (!f.isPersistent())
          continue;

        if (f.getName() != null && f.getName().Length>0)
          mFields.Add(f.getName(), f.getString());
      }

      // Get the stack (visited pages) and save as well
      // into "smartletStack" key
      ISmartletPage[] pages = smartlet.getHistory();
      String stack = "";
      for(int i=0;i<pages.Length;i++) {
      ISmartletPage page = pages[i];
        // Get page id
        stack = stack+ page.getId()+ ":";
      }
                  // Add the current page id on top of that
      stack = stack+ smartlet.getCurrentPage().getId();
      mFields.Add("smartletstack", stack);

      // serialize and save to file
      try {
          Stream s = File.Open(fileName, FileMode.Create,
                FileAccess.ReadWrite);
          BinaryFormatter b = new BinaryFormatter();
          b.Serialize(s, mFields);
      } catch (IOException e) {
          Console.WriteLine(e.Message);
      } finally {
        try {
            s.Close();
        }
        catch (Exception e) { }
      }
    }
  }
}
```

- Ensuite créez la classe **SmartletFieldFindAll.cs**:

```
using System;
using System.Collections.Generic;
using System.Text;
using com.alphinat.sg5;
using System.Collections;

namespace Training
{
  public class SmartletFieldFindAll : ISmartletElementVisitor {
    private bool found = false;
    private ArrayList fList = new ArrayList();

    public SmartletFieldFindAll() {
    }

    public ArrayList getTarget()
    {
      return fList;
    }

    public bool visit(ISmartletElement field) {
      return true;
    }

    public bool visitEnter(ISmartletElement element) {
      if (element is ISmartletField){
        ISmartletField field = (ISmartletField)element;
        fList.Add(field);
      }
      return true;
    }

    public bool visitLeave(ISmartletElement field) {
      return !found;
    }
  }
}
```

Créez le fichier descriptif XML

Le fichier *apn-extension-function.xml* représentant la classe *Utils* expose la fonction *saveToFile* dans SmartGuide Designer. Parce que cette fonction prend un nom de fichier en paramètre, celui-ci doit être déclaré explicitement dans le fichier descriptif.

- Ajoutez à votre projet un fichier **apn-extension-function.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<classes>
  <class>
    <name>MyBusiness.Utils</name>
    <methods>
      <method>
        <name>SaveToFile</name>
        <parameters>
          <parameter>
            <name>FileName</name>
          </parameter>
        </parameters>
      </method>
    </methods>
  </class>
</classes>
```

Compilez la fonction d'extension

Compilez votre projet pour générer la dll de librairies de classe.

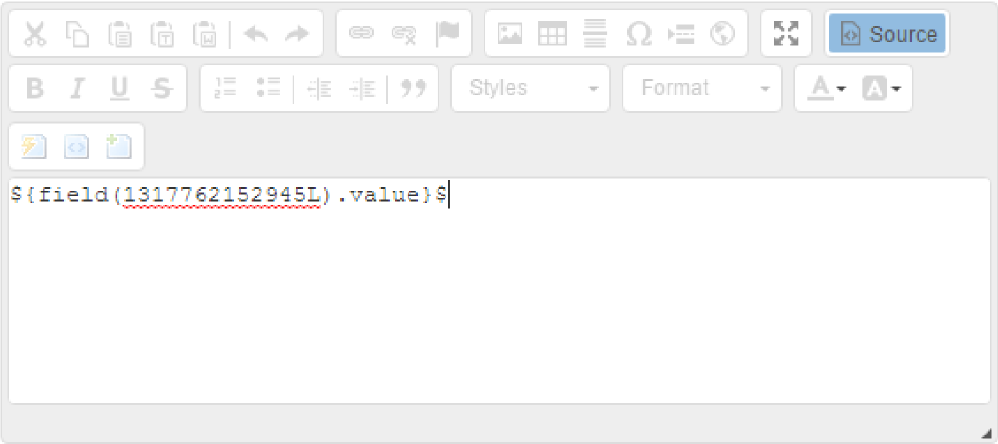
Reste ensuite à faire appel à la fonction d'extension à partir de SmartGuide Designer. Veuillez vous référer à la section 11, *Appels à des services* du guide de l'utilisateur SmartGuide Designer pour des instructions détaillées.

BeanShell

5

Introduction

BeanShell (http://www.beanshell.org) est un interpréteur de source Java. Il est inclus dans SmartGuide afin de permettre l'évaluation d'expressions avancées ou des validations de pages avancées par exemple. Cet interpréteur est également disponible dans les mappings de champs pour fichier PDF/XML ou les appels de service, ainsi que dans l'exécution de calculs. Par défaut il est caché dans SmartGuide Designer et peut être révélé en cliquant sur le bouton "source" dans l'éditeur d'expressions:



```
${field(1317762152945L).value}$
```

Passer en mode source est nécessaire lorsque vous souhaitez effectuer des manipulations plus avancées. Cette section du guide du développeur couvre le codage BeanShell, soit par les fonctions de l'API, ou grâce à des fonctions spécifiques fournies par SmartGuide dans le contexte d'exécution BeanShell.

Utiliser l'API V5

Coder dans BeanShell en utilisant l'API est essentiellement équivalent au codage d'une fonction d'extension en Java. Les principales différences sont les suivantes:

- la déclaration du type n'est pas nécessaire;
- il existe des raccourcis dans la syntaxe pour accéder des fonctions où le préfixe "get" est enlevé et la lettre qui suit mise en minuscule.

Comme exemple de ces différences, considérons la fonction suivante en BeanShell pour invoquer un service par son nom (appelé *Customer List* dans cet exemple):

```
${
  ISmartlet smartlet = context.smartlet;
  ISmartletService[] services = smartlet.getServices();
  ISmartletService service = null;
  for(int i=0;i<services.length;i++) {
    service = services[i];
    if ("Customer List".equals(service.getName())) {
      service.call();
      break;
    }
  }
}$
```

Le deuxième point est illustré sur la première ligne.

Comme autre exemple, le morceau de code suivant envoie l'utilisateur vers une autre page, appellée *Page 2* (et ajuste l'historique de navigation en conséquence):

```
${
  ISmartletPage currentPage = smartlet.getCurrentPage();
  ISmartletPage[] history = smartlet.getHistory();
  len = history.length+1;
  ISmartletPage[] newhistory = new ISmartletPage[len];
  for(int i=0;i<history.length;i++) {
    newhistory[i] = history[i];
  }
  newhistory[history.length] = currentPage;
  smartlet.setHistory(newhistory);
  ISmartletPage newPage = smartlet.findPageByName("Page 2");
  smartlet.setCurrentPage(newPage);
}$
```

La spécification du type, bien qu'optionnelle tel qu'illustré sur la ligne 3, est pratique parfois pour mieux comprendre le code.

Fonctions et extensions fournies par SmartGuide

Il y a aussi des objets prédéfinis et des fonctions pour effectuer des manipulations qui font usage de l'ancien API. Voici un exemple qui montre comment obtenir un paramètre de l'URL et mettre à jour la valeur du champ en cours avec elle:

```
${
  previd = field("hidCurrentCustomerId").value;
  custid = "";
  if (env.getAttribute(7, "id") != null) {
    custid = (String)((List)env.getAttribute(7, "id")).get(0);
  }
  if (custid != null && custid.length()
```

Le champ est appellé *hidCurrentCustomerId*, et le paramètre d'URL est *id*. Noter l'usage de la variable intrinsèque *env* ainsi que de la commande *field*. Dans les paragraphes suivants, nous présentons la liste des commandes et des objets disponibles dans les scripts BeanShell.

Une extension est également fournie laquelle supporte une syntaxe de liste comme suit,

```
${
        return ["mtl","tor",["or"]];
}$
```

Dans cet exemple une liste avec 3 elements de premier niveau est définie. Le troisième élément même est également une liste, contenant une seul élément. Cette syntaxe peut-être utile par exemple lorsqu'un service web requiert des données répétées dans ses paramètres d'entrée.

Référence du scripting dans les Smartlet

Ci-dessous une liste des fonctions intrinsèques disponibles dans l'environnement BeanShell SmartGuide.

Déclaration	Description

FieldInfo field(string | long)	obtient FieldInfo par nom de champ ou id de champ (long) ex.
	`field("text1");`
	`field(1234567890L);`

ServiceInfo service(string | long)	obtient ServiceInfo par nom de service ou identifiant de service (long) e.g.
	`service("web-service1");`
	`service("ext-function-call");`
	`service(1234567890L);`

int position()	retourne la position courante lorsqu'à l'intérieur d'un groupe répétable. L'index commence à 0.
APNOption option(string libelle, string valeur)	retourne un objet APNOption avec le libelle et la valeur spécifiée. Peut être utilisé dans un mapping de retour vers un champ à valeurs multiples, permettant d'assigner séparément les libellés et valeurs d'options, `option(service.getOutput("country/name"), service.getOutput("country/code"));`

Objects

Il y a quatre variables prédéfinies disponibles dans l'environment SmartGuide BeanShell:

- "env3": correspond à l'environment API V3 (com.alphinat.sg.Environment)
- "context": correspond contexte se service API V5 (com.alphinat.sg5.IServiceContext)
- "smartlet": correspond à l'objet Smartlet API V5 (com.alphinat.sg5.ISmartlet)
- "env": correspond à l'environment API V5 (com.alphinat.sg5.ISmartletEnvironment)

Prendre note que l'objet "smartlet" peut agir comme fonction, en prenant un code de Smartlet en paramètre. Ceci permet le référencement de champs provenant d'autres Smartlets comme suit:

```
${
        userid = smartlet("licenseRegistration").field("userid").value;
}$
```

Il y a également quatre objets détaillés dans les sous sections suivantes: FieldInfo, SubSmartlet, APNDate, et ServiceInfo.

FieldInfo

Fonction	Description
getId	String getId(); //obtient l'identifiant de champ
getName	String getName(); //obtient le nom du champ

getType	String getType(); //obtient le type de champ
	Valeur retournée:
	• "button" - bouton
	• "check" - case à cocher
	• "date" - date
	• "drop" - liste déroulante
	• "group-begin" - début de groupe
	• "group-end" - fin de groupe
	• "hid" - champ caché
	• "lbox" - listbox
	• "num" - champ numérique
	• "pass" - mot de passe
	• "radio" - bouton radio
	• "staticImg" - image statique
	• "staticText" - texte statique
	• "sub-interview" - champ sous Smartlet
	• "text" - champ d'entrée texte
	• "textLong" - textarea

- "upload" - champ téléchargement

getValue

Object getValue(); //obtient la valeur d'un champ

Selon le type de champ, l'objet retourné peut être:

- String - pour les champs: check, drop, hid, lbox, pass, radio, staticImg, staticText, text, textLong et upload
- APNDate - pour le champ: date
- Number - pour le champ: num; peut être un Long ou Double. Si le champ est vide, retourne Long(0);
- SubSmartlet - pour les champ: subsmartlet

getString

String getString(); //obtient la valeur chaîne de caractère d'un champ

Selon le type de champ, la chaîne retournée peut être:

- Cette méthode renvoie la chaîne réelle stockée dans le champ.
- Pour un champ date ou numérique, retourne la date formatée ou un chiffre.
- Pour les sous Smartlet, retourne une chaîne xml représentant le sous Smartlet.
- Pour un champ de téléchargement, en fonction de votre configuration, ce sera soit un chemin vers le fichier local ou une chaîne encodée base64 représentant le contenu binaire du fichier.

setValue

void setValue(Object obj); //assigne la valeur du champ

Selon le type de champ, la valeur peut être:

- Pour les champs: check, drop, hid, lbox, pass, radio, staticImg, staticText, text, textLong et upload c'est la fonction obj.toString() qui est appellée afin de déterminer la valeur qui sera assignée au champ. Si obj est nul, le champ est réinitialisé.
- Pour les champs: date, obj peut être une chaîne de caractères, le format accepté étant "yyyy-MM-dd", ou encore obj peut être de type Date. Si obj est nul, le champ est réinitialisé.
- Pour les champs: number obj doit être un chiffre.
- Pour les subsmartlet: non implémenté.

setString

void setString(String val); //assigne une valeur texte

Cette fonction assigne la valeur texte au champ.

EXCEPTION: lorsque l'on applique *setString* à un champ date, le format doit être consistent avec le format défini pour ce champ dans l'onglet de validation.

getValues	Object[] getValues(); //obtient les valeurs d'un champ répété
	Cette fonction retourne un tableau d'objets. Veuillez réferer à *getValue* pour chaque type d'objet. Si le champ n'est pas partie d'un groupe répétable, un tableau contenant un seul objet sera retourné.

getStrings	String[] getStrings(); //obtient le contenu texte d'un champ répété
	Cette méthode retourne un tableau de String. Se référer à *getString pour plus de détails. Si le champ n'est pas partie d'un groupe répétable, un tableau contenant une seule chaîne sera retourné.*

getValue(int)	Object getValue(int index); //obtient la valeur du champ répété à la position "index"
	Se référer à *getValue* pour chaque type d'objet. Si l'index est hors-limite, retourne null.

getString(int)	String getString(int index); //obtient le contenu texte du champ répété à la position "index"
	Se référer à *getString* pour chaque type d'objet. Si l'index est hors-limite, retourne null.

setValues	void setValues(Object[] values); assigne les valeurs à un champ répété Se référer à *getValue* pour chaque type d'objet.
setStrings	void setStrings(Object[] values); assigne des chaînes de caractères à un champ répété Se référer à *getString* pour chaque type d'objet.

Fonctions spécifiques aux champs de type liste: checkbox, dropdown, listbox, radio button

Si vous appelez les méthodes suivantes sur un champ autre que : check, drop, lbox et radio, une exception de type "Method/Attribute not found Exception" sera lancée.

getOptions	SelectionItem[] getOptions(); //obtient la liste des valeur
	Se référer à la section de référence de l'API pour le type de retour *SelectionItem*.

getSelectionItemList	SelectionItemList getSelectionItemList(); //obtient le SelectionItemList
	Cette fonction retourne un *SelectionItemList*. Un changement dans la liste des éléments aura une incidence sur les éléments de la liste. Se référer à la section de référence de l'API pour le type de retour *SelectionItemList*.

getOption(int)	SelectionItem getOption(int index); //obtient un SelectionItem selon l'index spécifié
	Une exception "Index out of bounds Exception" sera lancée si l'index est en dehors des limites de la liste.

getSelectionItem(int)	SelectionItem getSelectionItem(int index); //obtient un SelectionItem basé sur l'index
	Cette fonction est la même que *getOption(int)*.

getSelectedOptions()	SelectionItem[] getSelectedOptions(); //obtient les éléments sélectionnés sous forme de tableau
	Si aucune option n'est sélectionnée, null sera renvoyé au lieu d'un tableau vide. Pour des boutons radio ou une liste déroulante, si un élément est sélectionné, un tableau avec un élément sera retourné. Pour des cases à cocher ou un listbox, le tableau retourné peut contenir plus d'un élément selon le choix de l'utilisateur.

getSelectedLabels()	String[] getSelectedLabels(); //obtient les valeurs des éléments sélectionnés sous forme de tableau
	Si aucune option n'est sélectionnée, null sera renvoyé au lieu d'un tableau vide. Pour des boutons radio ou une liste déroulante, si un élément est sélectionné, un tableau avec un élément sera retourné. Pour des cases à cocher ou un listbox, le tableau retourné peut contenir plus d'un élément selon le choix de l'utilisateur.

getSelectedOption()	SelectionItem getSelectedOption(); //obtient le premier élément sélectionné
	Pour une liste déroulante ou des boutons radio, retournera null (pas de sélection), ou encore l'élément sélectionné. Pour des cases à cocher ou un listbox, le premier élément sélectionné sera retourné.

getSelectedLabel()	String getSelectedLabel(); //obtient le premier libellé d'élément sélectionné
	Pour une liste déroulante ou des boutons radio, retournera null (pas de sélection), ou encore le libellé de l'élément sélectionné. Pour des cases à cocher ou un listbox, le libellé du premier élément sélectionné sera retourné.

Fonctions spécifique au champ de type téléchargement

Si les méthode suivantes sont appellées sur un champ autre que téléchargement, une exception de type "Method/Attribute not found Exception" sera lancée.

getFileName()	String getFileName(); //obtient le nom du fichier téléchargé

getFileNames()	String[] getFileNames(); //obtient un tableau des noms de fichiers téléchargé pour un champ répété
	Si le champ ne fait pas partie d'un groupe répétable, un tableau contenant une seule chaîne sera retourné.

getFileSize()	long getFileSize(); //obtient la grosseur du fichier téléchargé

getFileSizes()	long[] getFileSizes(); //obtient les grosseurs des fichiers téléchargés pour un champ répété
	Si le champ ne fait pas partie d'un groupe répétable, un tableau contenant un seul nombre (long) sera retourné.

SubSmartlet

Method	Description

field(long|String)	FieldInfo field(long|String); //obtient un sous Smartlet par identifiant ou nom
	Cette fonction est similaire à la commande *field* mais dans un contexte de sous Smartlet.

getElement()	Element getElement(); //Obtient l'élément Dom4J d'un champ sous Smartlet
	Retourne nul si le champ est vide.

toString()	String toString(); //Obtient une chaîne XML pour le champ sous Smartlet
	Retourne une chaîne vide si le champ est vide.

APNDate

Cette classe est une sous classe de *java.util.Date* avec quelques méthodes additionnelles.

Fonction	Description
getYear()	int getYear();

getMonth()	int getMonth(); Le premier mois de l'année est janvier avec la valeur 0.
getDay()	int getDay(); Le premier jour du mois a la valeur 1.
setYear(int)	setYear(int year);
setMonth(int)	void setMonth(int month); Le premier mois de l'année est janvier avec la valeur 0.
setDay(int)	void setDay(int day); Le premier jour du mois a la valeur 1.
format(String)	String format(String format); Retourne une chaîne formattée du champ date selon le paramètre *format*. Consulter le guide de l'utilisateur SmartGuide Designer pour une liste d'exemples de formats de date.
setDate(int, int, int)	void setDate(int year, int month, int day);
setDate(String)	void setDate(string); La chaîne doit avoir le format: "yyyy-MM-dd"

ServiceInfo

Method	Description
call()	boolean call() throws Throwable; Appelle un service web ou une fonction d'extension définie dans le Smartlet.
getOutput(String xpath)	Object getOutput(String xpath); Obtient le retour d'un appel de service. Pour les appels de service Web, ce retour sera une chaîne de caractère basée sur le langage XPath. Si rien n'est trouvé, null sera retourné. Pour les appels de fonction

	extension, il peut y avoir un retour ou pas. Le paramètre *XPath est ignoré.*
getError()	String getError(); Obtient le message d'erreur retorné par l'appel de service.

Smartlets exposés en services web REST

6

Introduction

Une des caractéristiques de SmartGuide Server 5.1.0 et plus est la capacité d'être appelé non seulement via une interface web (à travers le rendu web normal d'une application) mais également de façon programmatique via une interface service web de type REST. En d'autres mots, l'URL du Smartlet peut être appelée avec des paramètres spécifiques afin de fournir des données à l'application, incluant la capacité de se déplacer à travers les pages de l'application, et recevoir un rendu en format JSON des pages résultantes ou des champs. Ceci permet par exemple le développement d'applications internet riches (RIA) en utilisant SmartGuide, et facilite également l'intégration de SmartGuide avec des systèmes externes.

Cette section décrit comment configurer le serveur SmartGuide pour activer cette fonctionnalité, ainsi que la liste des URL et paramètres disponibles pour appeler l'application dans ce mode.

Configuration de SmartGuide Server

Pour exposer les Smartlets sous forme de service web REST sous la version J2EE de SmartGuide, vous devez modifier le fichier de configuration *web.xml* sous **smartlets.war/WEB-INF** et décommenter les lignes suivantes:

```
<!--
<servlet>
<servlet-name>ajax-json-provider-servlet</servlet-name>
<servlet-class>com.alphinat.interview.si.xml.servlet.XMLServlet</servlet-class>
<init-param>
<param-name>config</param-name>
<param-value>/WEB-INF/config/ajax-json-provider-config.xml</param-value>
</init-param>
</servlet>
-->
<!-- Expose smartlet as AJAX json -->
<!--
<servlet-mapping>
<servlet-name>ajax-json-provider-servlet</servlet-name>
<url-pattern>/ajax/json/*</url-pattern>
</servlet-mapping>
-->
```

Pour la version .Net de SmartGuide cette fonctionalité est activée par défaut. Elle peut être désactivée en enlevant le fichier *web.config* sous le répertoire **ajax\json** de SmartGuide Server.

Utilisation

Il ya trois types d'objets que l'on peut obtenir: le Smartlet, une page du Smartlet, et un champ d'une page.

Obtenir l'objet JSON du Smartlet

Vous pouvez utiliser l'URL suivante pour accéder au Smartlet:

- **J2EE**
  http://host/smartlets/ajax/json/[process]/smartlet

- **.Net**
  http://host/smartlets/ajax/json/do.aspx/[process]/smartlet

  où la portion [process] de l'URL est optionnelle. S'il est spécifié, le serveur SmartGuide va traiter les paramètres postés.

Paramètres d'URL optionnels:

- ContentType:
  Utilisé pour spécifier le type de contenu, Ex. 'text/plain'. Si omis, le type 'application/json' sera retourné.
- filters :
  filtres d'objets, voir "filtres" ci-bas pour plus de détails

Obtenir l'objet JSON de la page

Vous pouvez utiliser l'URL suivante pour accéder au Smartlet:

- **J2EE**
  http://host/smartlets/ajax/json/[process]/page

- **.Net**
  http://host/smartlets/ajax/json/do.aspx/[process]/page

  où la portion [process] de l'URL est optionnelle. S'il est spécifié, le serveur SmartGuide va traiter les paramètres postés.

Paramètres d'URL optionnels:

- ContentType :
  Utilisé pour spécifier le type de contenu, Ex. 'text/plain'. Si omis, le type 'application/json' sera retourné.
- filters :
  filtres d'objets, voir "filtres" ci-bas pour plus de détails

- name :
  nom de la page désirée
- id :
  identifiant de la page désirée

Si aucun des paramètres "name" ou "id" n'est passé, alors la page courante du Smartlet est retournée.

Obtenir l'objet JSON d'un champ

Vous pouvez utiliser l'URL suivante pour accéder au Smartlet:

- **J2EE**
  http://host/smartlets/ajax/json/[process]/field

- **.Net**
  http://host/smartlets/ajax/json/do.aspx/[process]/field

  où la portion [process] de l'URL est optionnelle. S'il est spécifié, le serveur SmartGuide va traiter les paramètres postés.

Paramètres d'URL optionnels:

- ContentType :
  Utilisé pour spécifier le type de contenu, Ex. 'text/plain'. Si omis, le type 'application/json' sera retourné.
- filters :
  filtres d'objets, voir "filtres" ci-bas pour plus de détails
- name :
  nom du champ désiré
- id :
  identifiant du champ désiré
- htmlName :
  nom html du champ désiré, ex: d_123456789

Filtres

Si aucun paramètre "filtres" n'est donné pour l'appel ajax, toutes les informations détaillées seront retournées. Pour éviter un énorme objet JSON pour les Smartlets complexes vous pouvez choisir de filtrer quelques informations non nécéssaires.

Filtre	Description
"notavailable"	Filtre les éléments non disponibles. Les scénarios suivantes sont considérés comme non disponibles: -La disponibilité (visibilité) du champ est calculée et est

	fausse.(Si la disponibilité du champ n'est pas calculée, ce champ est considéré comme disponible.) -La page ne fait pas partie de l'historique de navigation où n'est pas la page courante Les éléments filtrés ne sont pas présent dans l'objet JSON retournés.
"subfields"	Filtre les sous éléments des champs de type groupe et groupe répétable. Les éléments filtrés sont présentés comme des objets vides, et par conséquent la longueur du tableau retourné est exacte
"subsmartletdetail"	Filtre le détail d'un sous Smartlet. Les éléments filtrés sont présentés comme des objet vides ou nuls. Par conséquent la valeur peut être utilisée pour vérifier si le sous Smartlet est vide ou non.
"summarydetail"	Filtre de l'information sur le sommaire. L'attribut "pages" est présenté comme un simple tableau d'objet page sans les attributes de champ de sorte que l'information de base de l'historique est accessible.

La portion serveur de SmartGuide Server va vérifier la présence du paramètre spécifié et activer le filtre.

Ex.

http://localhost/smartlets/ajax/json/smartlet?filters=subsmartletdetail_summarydetail

Exemples d'objets JSON

Il ya trois types d'objets que l'on peut obtenir: le Smartlet, une page du Smartlet, et un champ d'une page.

Objet JSON Smartlet:

```
{"smartlet":
 {
 "id" : "1308250108692", //ID du Smartlet
 "type" : "smartlet", //Type d'objet est "smartlet"
 "code" : "smartlet_code", //Code du Smartlet
 "name" : "smartlet name", //Nom du Smartlet
 "subject" : "", //Sujet du Smartlet
 "author" : "", //Auteur du Smartlet
 "description" : "", //Description du Smartlet
 "keywords" : "", //Mots-clés du Smartlet
"currentPageId" : "1308250108693", //ID de la page courante
"history" : [...], //Tableau des ID des pages visitées
"globalNavButtons" : [...], //Boutons de navigation globale
"pages" : [...], //Tableau d'objet de page JSON
"theme" : "default" //Thème
"layout" : "bootstrap" //Modèle de disposition des champs utilisé par le Smartlet
"layoutdefaultsize": "md" //Dimension par défaut pour la disposition des champs
"locales" : ["en", "fr"] //Tableau des locales en usage par le Smartlet
"lastmodification" : "1455831011" //Dernière date de modification du Smartlet
}
}
```

Objet JSON page:

```
{"page":
{
"id" : "1308250108693", //ID de page
"type" : "page", //Le type d'objet est "page"
"name" : "page_name", //Nom de page
"title" : "Page title", //Titre de page
"progress" : "0", //Pourcentage de progrès.
"template" : "", //Gabarit de page
"state" : "state1", //État de la page
"fields" : [...], //Tableau d'objets champ JSON
"layout" : {...}, //Object composé de rangées et colonnes représentant la disposition des champs sur la page
"navNextButton" : {...}|undefined, //Si présent, objet du bouton de navigation suivant
"navPreviousButton": {...}|undefined,//Si présent, objet du bouton de navigation précédent
"navSummaryButton": {...}|undefined, //Si présent, objet du bouton de navigation au sommaire
"modifyPageButton": {...}|undefined, //Si présent, objet du bouton de navigation de modification d'une section dans un sommaire
"returnButton" : {...}|undefined, //Présent lorsque le Smartlet courant est un sous Smartlet, retourne le bouton du Smartlet principal.
"returnWithoutSaveButton" : {...}|undefined, //Présent lorsque le Smartlet courant est un sous Smartlet, retourne le bouton du Smartlet principal mais sans le bouton de sauvegarde.
"errorMessages" : [...]|undefined, //Présent lorsqu'une erreur de validation de page survient, retourne un tableau de messages d'erreur.
"errorCodes" : [...]|undefined, //Présent lorsqu'une erreur de validation de page survient, retourne un tableau de codes d'erreur (entiers).
"smartlet" : {id:"", code:"", name:"", keywords:"", subject"", "type":"", "theme":"" }, //Information sommaire du Smartlet
}
}
```

Objet JSON champ:

```
{
"id" : "1308250108694", //ID du champ
"type" : "text", //Type de champ,
//peut être: text,textLong,num,pass,date,upload,staticText,staticImg,
//radio,check,drop,lbox, button,hid,
//knowledge,sub-smartlet,group,repeat,summary
"typeDetail" : "", //Détail du type de champs, peut être:
// button_subsmartlet_enter,button_subsmartlet_return_save,button_subsmartlet_without_save,
// button_next_page, button_previous_page, button_repeat_insert,button_repeat_delete,
// button_goto_summary,button_modify_page,button_refreh_page,button_global_navigation
// button_gen_pdf,button_gen_xml
"name" : "field_name", //Nom du champ
"htmlName" : "d_1308250108694", //Nom html du champ
"label" : "field label", //Libellé du champ
"value" : "", //Valeur texte du champ
"help" : "field contextual help", //Aide contextuelle
"isHelpLink" : true|false, //true si le texte d'aide est un lien
"tooltip" : "field tooltip", //Tooltip
"format" : "", //Format
"isValid" : true|false|undefined, //est-ce que le champ est valide, non calculé si non défini
"isAvailable" : true|false|undefined, //est-ce que le champ est disponible, non calculé si non défini
"isRequired" : true|false, //est-ce que le champ est requis
"maxLength" : -1, //longueur de champ maximale
"minLength" : -1, //longueur de champ minimale
"cssClass" : "", //classe css
"cssStyle" : "", //style css
"cssHeight" : "", //hauteur css
"cssWidth" : "", //largeur css
"isReadonly" : true|false, //indicateur readonly
"isPersistent" : true|false, //indicateur de persistence
"isEncrypted" : true|false, //indicateur d'encryption
"suffix" : "", //suffixe du champ
"prefix" : "", //préfixe du champ
"choiceLayout" : "", //présentation des éléments pour champs de type liste
"eventtarget" : "", //liste des id de champs ayant une dépendance sur le champ courant
"eventsource" : "", //liste des id de champs ayant un impact sur le champ courant
"layout" : [...], //tableau des propriétés de disposition par type d'appareil
////Les attributs suivants sont pour les champs de type liste : radio,check,drop,lbox
"options" : [{
"isOptionGroup" : true|false, //option possède des sous options
"label" : "", //libellé de l'option
"value" : "", //valeur de l'option
"help" : "", //aide de l'option
"isHelpLink" : true|false, //est-ce que l'aide est un lien
"hint" : "", //indice sur l'option
"subOptions" : []|undefined //sous options si "isOptionGroup" est égal à true
},...], //tableau des objet d'option
////Les attributs suivants sont pour le champ groupe répétable
"groups" : [...], //groupes répétés
"addButton" : {...}, //bouton d'ajout d'instance de groupe
"deleteButton" : {...}, //bouton de suppression d'instance de groupe
////Les attributs suivants sont pour les champs de type téléchargement
"fileName" : "", //nom du fichier téléchargé
////Les attributs suivants sont pour les boutons:button_gen_pdf,button_gen_xml
"genFileName" : "", //nom du fichier pdf ou xsd
////Les attributs suivants sont pour les éléments des bases de connaissances
"knowledgeEntries" : [{ //entrée de base de connaissance, l'élément est une entrée
"name" : "", //nom de l'entrée
"label" : "", //libellé de l'entrée
"value" : "", //valeur de l'entrée
"isRepeat" : true|false, //si répétée
"entries" : [...] //sous entrées
}...],
////Les attributs suivants sont pour SessionSummary
"modifyButtonLabel" : {}, //libellé pour le bouton modifier
"summaryButtonLabel" : {}, //libellé pour le bouton aller au sommaire
"pages" : [...], //tableau des pages pertinentes, l'élément est un objet de page JSON.
//Si le filtre summarydetail est présent, l'élément est un objet sommaire de page avec les attributs: id,name,title,type,state,template
////Les attributs suivants sont pour le champ de type groupe
"fields" : [...], //tableau des sous champs du groupe, l'élément est un objet de champ JSON.
////Les attributs suivants sont pour un champ de type sous Smartlet
"subSmartletCode", : "", //code du sous Smartlet
"enterButton", : {...}, //bouton d'entrée du sous Smartlet
"subSmartlet", : {} //Objet sous Smartlet JSON. Est un objet vide si le filtre subsmartletdetail est présent
}
```

Pratiques recommandées et exemple

Pratique recommandées

Dans la plupart des cas, vous avez seulement besoin de traiter puis d'effetuer le rendu de la page actuelle du Smartlet. Un appel AJAX à *http://localhost/smartlets/ajax/json/process/page* va retourner suffisamment d'information pour effectuer le rendu du Smartlet.

Habituellement le premier appel AJAX est: *http://localhost/smartlets/ajax/json/process/page?interviewID=smartletCode&filters=notavailable*. Cet appel permet au Smartlet de s'initialiser.

Si vous avez besoin de télécharger un champ, la valeur recommandée pour *ContentType* est *ContentType=text/plain*, sinon une boite de dialogue "Sauvegarde sous" pourrait être présentée par le navigateur.

Lors de la soumission du form (via AJAX), l'action du form est babituellement:
http://localhost/smartlets/ajax/json/process/page.

Toujours fournir le paramètre *filters=notavailable* si vous avez seulement besoin de faire un rendu de la page coutante avec tout les éléments disponibles.

Exemple

Dans cet exemple nous supposons l'existence d'un Smartlet de 3 pages dont le code est *Contact*. Nous allons obtenir le Smartlet, populer un champ *name* sur la page 1, aller à la page 2, populer un champ *address*, et finalement arriver en page 3 où nous allons obtenir le contenu de la page.

La séquence d'appels est la suivante:

- Initialiser et obtenir le Smartlet:
  http://localhost/smartlets/ajax/json/process/page?interviewID=Contact&filters=notavailable
- Poster la valeur du champ name et déclencher le passage à la page suivante:
  http://localhost/smartlets/ajax/json/process/page?d_1324588352621=John&t_n1324588352620=t_n1324588352620&filters=notavailable
- Poster la valeur du champ address et déclencher le passage à la page suivante:
  http://localhost/smartlets/ajax/json/process/page?d_1324588352623=123%20main&t_n1324588352622=t_n1324588352622&filters=notavailable

Dans les deuxième et troisième appels du "Smartlet" le nom des champs html a été utilisé. Ces noms sont extraits de la réponse JSON pour les premier et deuxième appels. Vous devez analyser la réponse JSON et obtenir la propriété "htmlName" du champs que vous souhaitez poster. Pour le bouton de navigation pour accéder à la page suivante, le champ que vous devez analyser est "navNextButton".

Meilleures pratiques

7

Introduction

Le but de ce document est de fournir des recommandations quant à l'usage de services web, de fonctions d'extension et de code « BeanShell » dans le développement d'applications avec SmartGuide. En particulier ce document adresse les circonstances dans lesquelles il est recommandé d'utiliser une approche ou une autre pour exécuter des fonctions externes à SmartGuide.

Utilisation des services web

Étant donné que SmartGuide Designer permet de consommer des services web de façon native sans programmation, cette pratique est la plus recommandée pour communiquer avec des services externes.

En particulier, voici une liste non limitative de circonstances dans lesquelles nous recommandons l'utilisation directe de services web à partir de SmartGuide :

- Des services web existent déjà au sein de l'entreprise pour effectuer la tâche requise;
- Des services web n'existent pas au sein de l'entreprise, mais les fonctionnalités requises pourraient aussi être utilisées par d'autres systèmes au sein de l'entreprise non basés sur la technologie SmartGuide;
- Pour pré-remplir des listes dynamiques tel que des listes déroulantes, boutons radios, cases à cocher;
- Pour effectuer une validation de champ ou page;
- Pour transférer des données provenant d'un ou plusieurs champs vers un système externe (Ex : sauvegarde de données, transmission des extrants d'un parcours, etc.);
- Pour déclencher une action dans un système externe (Ex : activer un « workflow », effectuer une transaction, etc.);
- Pour charger dans données dans un parcours à partir d'un système externe (Ex : récupérer des données personnelles sur un utilisateur, charger des paramètres de configuration, initialiser la valeur de certains champs).

Utilisation de fonctions d'extension

Dans certains cas, il peut être souhaitable d'effectuer les appels à des services web par l'intermédiaire d'une fonction d'extension plutôt que directement à travers SmartGuide. Ci-après, une liste non exhaustive de circonstances où l'appel d'un service à travers une fonction d'extension est requis ou souhaitable:

- L'objectif final de l'appel du service web requiert plusieurs appels subséquents au même web service selon les valeurs retournées par le premier appel;

- Le service web retourne un grand nombre de champs et seulement quelques-uns sont nécessaires dans la réponse souhaitée. Dans ce cas-ci, on voudra à tout le moins ne pas utiliser l'option "Auto-créer les champs à partir du service";
- Le service web retourne de types de données complexes (ou prend en entrée des types complexes) qui ne peuvent pas être interprétés par SmartGuide (Ex : des types d'objets spécifiques à l'entreprise);
- Pour effectuer une validation de champ ou de page;
- Le service web est appelé très souvent comme pour remplir une liste de villes dans une liste déroulante dans plusieurs pages différentes. Dans ce cas, il est souvent préférable d'appeler le service dans une fonction d'extension et de "cacher" l'information retournée dans la portée "application" ou "session" pour optimiser la performance;
- Lorsque plusieurs services web distincts doivent être appelés pour remplir une fonction spécifique. Par exemple si une validation requiert l'appel à plusieurs services web, une fonction d'extension sera alors préférable.

Il y a aussi certains cas où il est absolument nécessaire d'utiliser une fonction d'extension, indépendamment de l'utilisation de services web, comme par exemple:

- Lorsqu'on doit jouer avec la pile des pages SmartGuide ("stack") directement (par exemple pour rediriger vers une page spécifique d'un Smartlet);
- Lorsqu'on doit retourner des données dans le flux http directement (retourner un flux binaire, rediriger vers une autre page web) ou encore interagir avec le contexte http (session, application, etc.);
- Lorsqu'on doit conditionner des séquences d'appels à plusieurs services. Par exemple, si des erreurs ont été détectées dans les premiers appels alors les services subséquents pourraient ne pas être appelés.
- Lorsqu'on doit gérer des cas de double-clic (qui passent nécessairement par une interaction avec les variables de session ou le contexte http);
- Lorsqu'on doit générer un document PDF ou XML (ex : si on veut l'envoyer par email ou dans un système externe).
- Lorsqu'on a besoin d'accéder à la plupart ou à toutes les données dans un Smartlet;

Utilisation du BeanShell

SmartGuide permet l'utilisation de code "BeanShell" pour effectuer différentes opérations avancées dans une application web. L'utilisation du "BeanShell" est possible dans les contextes suivants:

- Au niveau de la valeur des champs;
- Au niveau des mappages entre les champs d'entrée et de sortie d'un appel de service;
- Au niveau des mappages entre les champs SmartGuide et fichier PDF ou XSD;
- Sur la validation avancée des pages;

- Sur les actions de boutons;
- Sur initialisation de Smartlet, ainsi qu'en entrée et sortie de page;
- Au niveau du point d'arrimage ("end point") lors de l'importation d'un service web.

Règle générale, il est recommandé d'utiliser du « BeanShell » dans les circonstances suivantes:

- Lorsqu'aucune fonction SmartGuide ne permet de faire directement l'opération désirée nativement;
- Pour effectuer des manipulations simples sur la valeur des champs;
- Pour manipuler la valeur des champs dans un mappage de fichier PDF ou XSD;
- Pour manipuler les intrants et extrants d'un appel de service;
- Pour effectuer des opérations spécifiques à un champ ou un contexte précis qui n'est pas générique.

Bien que le "BeanShell" permette de faire des manipulations avancées et complexes, il est recommandé d'effectuer les opérations plus complexes dans des fonctions d'extensions. Noter aussi que le "BeanShell" permet de faire appel à un service tel que défini dans SmartGuide, donc de faire l'appel à un service web ou une fonction d'extension pour faire des traitements plus avancés.

Utilisation du Bean Shell vs l'utilisation d'une fonction d'extension

Comme pour les services web, la question se pose à savoir si l'on doit utiliser du "BeanShell" ou utiliser une fonction d'extension pour remplir une tâche spécifique. Règles générales, on préconisera l'approche d'une fonction d'extension dans les circonstances suivantes (sans limitations):

- L'opération à effectuer requiert plusieurs dizaines de lignes de code "BeanShell". Il est plus facile de maintenir et de déboguer du code contenu dans une fonction d'extension que dans du « BeanShell » au niveau de SmartGuide Designer;
- Lorsque l'opération à effectuer est de nature générique, c'est-à-dire qu'elle peut être réutilisée dans plusieurs contextes et dans différents Smartlets;
- Pour des considérations de performances. Car le premier appel à une fonction Beanshell provoque une compilation du code java lequel peut impacter la performance de façon non négligeable.

Finalement, dans les circonstances suivantes il est préférable d'utiliser du code BeanShell plutôt qu'un fonction d'extension:

- Lorsque SmartGuide ne permet l'appel directement à un service comme dans un mappage de champ avec un PDF, un XSD ou un service lui-même. Noter que si le mappage requiert des opérations longues et complexes, le code "BeanShell" du mappage peut faire appel à une fonction d'extension au besoin;

Référence API

8

Les pages suivantes présentent le détail des interfaces et méthodes disponibles avec de nombreux exemples.

Voici un tableau sommaire des interfaces. La classe de base pour l'API SmartGuide est com.alphinat.sg5. Prendre note que puisque cette documentation provient de la javadoc, celle-ci est en anglais uniquement.

Interface	Description
IServiceContext	IServiceContext is an interface representing a service call context. This is the entrance point for extension functions using the version 5 API. the user can get the current ISmartlet, ISmartletEnvironment and ISmartletEvent.
ISmartletEnvironment	Environment is an interface used to obtain the Smartlet application environment.
ISmartlet	ISmartlet is an interface representing a Smartlet.
ISmartletPage	ISmartletPage is an interface representing a Smartlet page along with the operations that may be performed on that page.
ISmartletField	ISmartletField is an interface representing a Smartlet field along with the operations that may be performed on that field.
ISmartletEvent	ISmartletEvent is an interface representing a Smartlet event, like a click on a button.
ISmartletService	ISmartletService is an interface representing a Smartlet service and covers extension functions, SOAP and REST web services.
ISmartletElementVisitor	Implements the Hierarchical Visitor Pattern to traverse Smartlet elements.
ISmartletElement	A Smartlet element can be a ISmartlet, ISmartletPage, ISmartletField, ISmartletService. It is often used in the context of the visitor pattern to

	filter out elements in a Smartlet for processing.
ISmartletActionError	ISmartletEvent is an interface representing a Smartlet action error object. It is used to wrap information about an error occuring when processing actions.
ISmartletSummary	ISmartletSummary is an interface representing a summary element.
ISubSmartletField	ISubSmartletField is an interface representing a subSmartlet.
ISmartletSelectField	ISmartletField is an interface representing a Smartlet select type field. These can be radio button, dropdown list, checkbox or listbox.
ISelectOptionList	ISelectOptionList is an interface representing the single or multiple select option list associated to a select type field. Each list item is an instance of ISelectOption. Manipulating the list directly affects the corresponding field's option item list. A field's option list may be obtained by using the ISmartletSelectField.getSelectOptions() functionality.
ISelectOption	ISelectOption is an interface representing a single or multiple selection list item. Individual item attributes may be manipulated. New list item instances can be created using the ISelectOptionList.createOption() functionality.
ISmartletRepeat	ISmartletRepeat is an interface representing a Smartlet repeat widget. A repeat widget can be thought of as a table. Each row of the table is a group. And each column of the table is a field in the group.
ISmartletKnowledgeEntry	ISmartletKnowledgeEntry is an interface representing a Smartlet knowledge entry.
ISmartletKnowledge	ISmartletKnowledge is an interface representing a Smartlet knowledge widget.
ISmartletGroup	ISmartletGroup is an interface representing a Smartlet group.

`ISmartletDate`	ISmartletDate is an interface representing a Smartlet date.
`ISmartletUpload`	ISmartletUpload is an interface representing a Smartlet upload field.
`Constants`	This is an interface representing all constants used in the API. It covers element types, error codes, file type, scope, and Smartlet events. The constant values are available here.

com.alphinat.sg5

Interface IServiceContext

```
public interface IServiceContext
```

`IServiceContext` is an interface representing a service call context.
This is the entrance point for extension functions using the version 5 API. From `IServiceContext` the user can get the current `ISmartlet`, `ISmartletEnvironment` and `ISmartletEvent`.

Method Summary

`Object`	**`getAPI3Environment`**`()` Gets the Environment of API version 3.
`ISmartletField`	**`getContextField`**`()` Gets context `field`.
`ISmartletEnvironment`	**`getEnvironment`**`()` Gets the Smartlet process `environment`.
`ISmartletEvent`	**`getEvent`**`()` Gets the context event.
`ISmartlet`	**`getSmartlet`**`()` Gets the current `Smartlet` .

Method Detail

getSmartlet

```
ISmartlet getSmartlet()
```

Gets the current `Smartlet` .

Returns:

current `Smartlet`

---

getEvent

`ISmartletEvent` **`getEvent`**`()`

Gets the context event.

Allows to get access to the field that triggered an event.

For example when a button is clicked, the following code will retrieve the button.

```
ISmartletEvent ise = context.getEvent();
ISmartletField button = (ISmartletField)ise.getSource();
```

Returns:

current `event`

See Also:

`ISmartletEvent.getSource()`

---

getContextField

`ISmartletField` **`getContextField`**`()`

Gets context `field`.

Is functionally equivalent to getting the event and then getting the source (field) of the event.

Returns:

current context `field`

See Also:

`getEvent()`

---

getEnvironment

`ISmartletEnvironment` **`getEnvironment`**`()`

Gets the Smartlet process `environment`.

Allows access to objects related to the execution environment, like Request and Response for the http context, as well as attributes (application, session, configuration, etc).

The following example shows how to get a parameter from the web.xml (or web.config) file using the `CONFIGURATION` attribute,

```
    ISmartletEnvironment env = context.getEnvironment();
    string dbDriver = (String)env.getAttribute(com.alphinat.sg5.Constants.Scope.CONFIGURATION,
"dbDriver");
```

Returns:

current Smartlet process `environment`

---

getAPI3Environment

`Object` **`getAPI3Environment`**`()`

Gets the Environment of API version 3.

The V3 environment is the entry point that gives access to the Smartlet information. It allows the reuse of legacy code inside a V5 extension function.

The following example provides a map of field/values for the Smartlet,

```
    com.alphinat.sg.Environment env3 = (com.alphinat.sg.Environment) context.getAPI3Environment();
    Dictionary map = (Dictionary)env3.getData().convertToFieldDictionary();
```

Note that it is also possible to get the V5 API from the V3 API. One can use the following calls on the env (V3) environment

```
    env.getConfiguration().get("context5")
```

will return IServiceContext of API5

```
    env.getConfiguration().get("smartlet5")
```

will return ISmartlet of API5.

Returns:

`com.alphinat.sg.Environment` of version 3.

---

com.alphinat.sg5

Interface ISmartletEnvironment

`public interface` **`ISmartletEnvironment`**

`Environment` is an interface used to obtain the Smartlet application environment.

Method Summary

`Object`	**`getAttribute`**`(int scope, Object key)` Obtains the value of the attribute located within the specified environment scope.
`Object`	**`getAttributes`**`(int scope)` Obtains the specified environment scope.
`Object`	**`getContext`**`()` Gets the context associated to the environment.
`Object`	**`getRequest`**`()` Gets the Request object associated to the environment.
`Object`	**`getResponse`**`()` Gets the Response object associated to the environment.
`Object`	**`getUserPrincipal`**`()` Gets the principal reference containing the name of the current user.
`bool`	**`isUserInRole`**`(string role)` Determines whether the current user is included in the specified logical role.
`void`	**`redirect`**`(string location)` Sends a temporary redirect response to the client using the specified redirect location URL.
`void`	**`removeAttribute`**`(int scope, Object key)` Removes an attribute from the specified environment scope.
`void`	**`setAttribute`**`(int scope, Object key, Object value)`

Associates a value to a specified attribute name within a given environment scope.

Method Detail

getContext

```
Object getContext()
```

Gets the context associated to the environment. Within a J2EE Servlet and JSP tag environment, the returned object is an instance of `javax.servlet.ServletContext`. Within a JSR-168 portlet environment, the returned object is an instance of `javax.portlet.PortletContext`. Within a .NET HttpHandler and web Control environment, the returned object is an instance of `System.Web.HttpContext`.

The following example demonstrates how to use the context, under J2EE environments, to get the session ID.

```
ISmartletEnvironment env = context.getEnvironment();
javax.servlet.jsp.PageContext pc = (javax.servlet.jsp.PageContext)env.getContext();
String sessionId = pc.getSession().getId();
```

Under a .Net environment, one would use the following.

```
ISmartletEnvironment env = context.getEnvironment();
System.Web.HttpContext ctx = (System.Web.HttpContext) env.getContext();
string sessionID = ctx.Session.SessionID;
```

Returns:

the context associated to the environment.

---

getRequest

```
Object getRequest()
```

Gets the Request object associated to the environment. Within a J2EE Servlet and JSP tag environment, the returned object is an instance of `javax.servlet.ServletRequest`. Within a JSR-168 portlet environment, the returned object is an instance of `javax.portlet.PortletRequest`. Within a .NET HttpHandler and web Control environment, the returned object is an instance of `System.Web.HttpRequest`.

The following example demonstrates, under a J2EE environment, how to retrieve the query string of the current url from the request object.

```
import javax.servlet.jsp.*;
ISmartletEnvironment env = context.getEnvironment();
HttpServletRequest request = (HttpServletRequest)((PageContext)env.getContext()).getRequest();
String qs = request.getQueryString();
```

Similarly under .Net one would use.

```
ISmartletEnvironment env = context.getEnvironment();
System.Web.HttpRequest request = (System.Web.HttpRequest)env.getRequest();
NameValueCollection qry = request.QueryString;
string[] keys = qry.AllKeys;
for (int i = 0; i < keys.Length; i++)
{
  Console.WriteLine(keys[i] + "->" + qry[keys[i]]);
}
```

Returns:

http request object.

---

getResponse

`Object **getResponse**()`

Gets the Response object associated to the environment. Within a J2EE Servlet and JSP tag environment, the returned object is an instance of `javax.servlet.ServletResponse`. Within a JSR-168 portlet environment, the returned object is an instance of `javax.portlet.PortletResponse`. Within a .NET HttpHandler and web Control evironment, the returned object is an instance of `System.Web.HttpResponse`.

The following example demonstrates how to return a binary stream for a generated PDF document called documentName.

```
ISmartlet smartlet = context.getSmartlet();
byte[] bDoc = smartlet.generateFile(Constants.FileType.PDF, "documentName");
HttpServletResponse response = (HttpServletResponse)context.getEnvironment().getResponse();
response.setContentType("application/octet-stream");
response.setHeader("Content-Disposition","attachment; filename="+"documentName");
response.getOutputStream().write(bDoc);
response.getOutputStream().flush();
response.getOutputStream().close();
```

Under a .Net environment the code would read as follows.

```
ISmartlet smartlet = context.getSmartlet();
byte[] bDoc = smartlet.generateFile(Constants.FileType.PDF, "documentName");
HttpResponse response = (HttpResponse)context.getEnvironment().getResponse();
response.ContentType = "application/octet-stream";
response.AddHeader("Content-Disposition","attachment; filename="+docName);
response.OutputStream.Write(bDoc,0,bDoc.Length);
response.OutputStream.Flush();
response.OutputStream.Close();
```

Returns:

http response object.

---

getAttribute

```
Object getAttribute(int scope,
                    Object key)
```

Obtains the value of the attribute located within the specified environment scope. The environment returns `null` if the attribute does not exist within the specified scope.

The following example shows how to get a parameter from the web.xml (or web.config) file using the `CONFIGURATION` attribute,

```
ISmartletEnvironment env = context.getEnvironment();
String dbDriver = (String)env.getAttribute(Constants.Scope.CONFIGURATION, "dbDriver");
```

Parameters:

`scope` - the scope within which to locate the attribute whose name corresponds to the specified `key`.
`key` - the `String` representing the the name of the attribute whose value is to be obtained.

Returns:

the `Object` representing the value of the attribute located within the specified environment scope or `null` if the attribute could not be located.

See Also:

`Constants.Scope`

---

getAttributes

```
Object getAttributes(int scope)
```

Obtains the specified environment scope.

The following example shows how to iterate through all posted data.

```
ISmartletEnvironment env = context.getEnvironment();
Map mParams = (Map)env.getAttributes(Constants.Scope.PARAMETER);
Iterator itrParams = mParams.keySet().iterator();
while(itrParams.hasNext())
{
   String pName = (String)itrParams.next();
   List pValue = (List)mParams.get(pName);
   System.out.println(pName+","+pValue.toString());
}
```

Under a .Net environment the code would read as follows.

```
ISmartletEnvironment env = context.getEnvironment();
Dictionary map = (Dictionary)env.getAttributes(DotnetConstants.Scope.PARAMETER);
foreach(KeyValuePair kvp in map )
{
   Console.WriteLine("Key = {0}, Value = {1}", kvp.Key, kvp.Value);
}
```

Parameters:

`scope` - the scope to obtain.

Returns:

the `Object` representing the specified environment scope.

See Also:

`Constants.Scope`

---

setAttribute

```
void setAttribute(int scope,
                  Object key,
                  Object value)
```

Associates a value to a specified attribute name within a given environment scope. Specifying a `null` value has the same effect as removing the attribute from the environment scope.

Parameters:

`scope` - the scope within which to place the `value` according to the specified `key`.
`key` - the `String` representing the name of the attribute according to which to associate the `value` within the specified `scope`.
`value` - the `Object` representing the value to associate to the `scope` according to the specified `key`.

See Also:

`Constants.Scope`

---

removeAttribute

```
void removeAttribute(int scope,
                     Object key)
```

Removes an attribute from the specified environment scope.

Parameters:

`scope` - the scope from which to remove the attribute whose name corresponds to the specified `key`.
`key` - the `String` representing the name of the attribute that is to be removed from the specified `scope`.

See Also:

`Constants.Scope`

---

getUserPrincipal

```
Object getUserPrincipal()
```

Gets the principal reference containing the name of the current user.

Returns:

an `Object` containing the name of the user making this request, or `null` if no user information could be determined.

---

isUserInRole

```
bool isUserInRole(string role)
```

Determines whether the current user is included in the specified logical role. If the current user cannot be determined, false is returned.

Parameters:

`role` - the `String` specifying the name of the role.

Returns:

`true` if the current user can be determined and is included in the specified logical role, `false` otherwise.

---

redirect

```
void redirect(string location)
```

Sends a temporary redirect response to the client using the specified redirect location URL. This method can accept relative URLs; The servlet container must convert the relative URL to an absolute URL before sending the response to the client. If the location is relative without a leading '/' the container interprets it as relative to the current request URI. If the location is relative with a leading '/' the container interprets it as relative to the servlet container root. After using this method, the response should be considered to be committed and should not be written to.

Parameters:

`location` - -the redirect location URL

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5

Interface ISmartlet

All Superinterfaces:

ISmartletElement

`public interface` **`ISmartlet`**
extends ISmartletElement

`ISmartlet` is an interface representing a Smartlet.

Method Summary

`void`	**`addActionError`**`(ISmartletActionError actionError)` Adds an action error object.
`void`	**`addActionError`**`(Object sourceObject, string error, string callStack)` Adds an action error element composed of a source object, the error message and a stack trace.
`void`	**`addLocalizedResource`**`(string locale, string key, string value)` Add a key/value pair to the translation resources
`ISmartlet`	**`addPageToHistory`**`(ISmartletPage page)` Adds given page to the history.
`void`	**`calculate`**`()` Recalculates the page.
`void`	**`clear`**`()` Calls clear on every page of this smartlet, recursively.
`void`	**`clearActionErrors`**`()` Clears all action error of this smartlet.
`ISmartletField`	

	createField(ISmartletField anotherField) Create a dynamic field from existing field, a unique ID will be assigned to the new field.
ISmartletField	**createField**(string fieldName, int fieldType) Create a dynamic field, a unique ID will be assigned to the new field.
ISmartletField	**createField**(string newFieldId, ISmartletField anotherField) Create a dynamic field from existing field.
ISmartletField	**createField**(string newFieldId, string fieldName, int fieldType) Create a dynamic field.
Object	**evalBSH**(string bsh) Evaluate Beanshell scripts.
ISmartletField	**findFieldById**(string id) Finds the first matching field by id
ISmartletField	**findFieldByName**(string name) Finds the first matching field by name.
ISmartletPage	**findPageById**(string id) Finds a page by id
ISmartletPage	**findPageByName**(string name) Finds page by name.
ISmartletPage	**findPageByState**(string state) Finds a page by state string.
ISmartletService	**findServiceByName**(string name) Finds the service by given name.
byte[]	**generateFile**(int type, string name) Generates a XML or pdf file uploaded to the Smartlet.
byte[]	**generatePDFWithDictionarypingData**(string pdfFile, Object dataDictionary, bool flatten, bool readonly) Generates a pdf file with the provided pdf mapping data.

ISmartletActionError[]	**getActionErrors**() Gets an array of errors for the current Smartlet.
string	**getCode**() Gets the Smartlet code as defined on the properties page of the Smartlet.
string	**getCurrentLocale**() Gets the current locale for the Smartlet
string	**getCurrentLocaleDescription**() Gets the current language for the Smartlet
ISmartletPage	**getCurrentPage**() Gets the current page.
ISmartlet	**getCurrentSmartlet**() Gets the current Smartlet.
string	**getDomain**() Gets current domain name
ISmartletField[]	**getGlobalNavButtons**() Gets the global navigation buttons.
ISmartletPage[]	**getHistory**() Gets the history pages navigated by the user.
string	**getId**() Obtains the unique internal identifier of the Smartlet.
string	**getKeywords**() Gets the Smartlet keywords as defined on the properties page of the Smartlet.
string[]	**getLocales**() Gets the array of locales supported by the Smartlet
string[]	**getLocalesDescription**() Gets the array of locales description (languages) supported by the Smartlet
string	**getLocalizedResource**(string key) Gets value corresponding to a custom key for the resources
string	**getName**()

	Gets the Smartlet name as defined on the properties page of the Smartlet.
ISmartletPage[]	**getPages**() Gets the pages of a Smartlet as an Array.
ISmartletField	**getParentSubSmartletField**() If the current Smartlet is a subSmartlet, gets the parent subSmartlet field.
Object	**getPDFDictionarypingData**(string pdfFileName) Gets the PDF mapping data.
string	**getProgress**() Returns the current percentage complete, from "0" to "100"
ISmartletService[]	**getServices**() Gets the services of the Smartlet.
string	**getSubject**() Gets the Smartlet subject as defined on the properties page of the Smartlet.
ISmartletField	**getSubSmartletCancelButton**() Gets the button to return from sub smartlet without save.
ISmartletField	**getSubSmartletReturnButton**() Gets the button to return from subsmartlet.
string	**getTheme**() Gets the theme name of Smartlet as defined on the properties page of the Smartlet.
string	**getWorkspace**() Gets current workspace name
ISmartletPage	**gotoPage**(long pageId) Navigate to page with given page id and add current page to history.
ISmartletPage	**gotoPage**(string pageName) Navigate to page with given page name and add current page to history.
ISmartletPage	**gotoPage**(string pageName, bool addCurrentPageToHistory) Navigate to page with given page name and add current page to

	history.
ISmartletPage	**gotoPage**(string pageName, bool addCurrentPageToHistory, bool callServicesOnPageExit, bool callServicesOnPageEntry) Navigate to page with given page name and add current page to history.
ISmartletPage	**gotoPage**(string pageName, bool preserveHistory, bool callServicesOnPageExit, bool callServicesOnPageEntry, string[] pageNamesToAddToHistory) Navigate to page with given page name.
ISmartlet	**gotoSmartlet**(string smartletCode, string destinationPageId, bool preserveHistoryAndData, bool callServicesOnPageExit) Navigate to Smartlet with given code.
bool	**hasPage**(string pageName) Check if this smartlet
bool	**isSubSmartlet**() Check if we are inside a subSmartlet.
void	**sendMail**(string from, string to, string cc, string bcc, string returnTo, string subject, string body, int format, string[] attachmentNames, byte[][] attachements) Send email.
void	**sendMail**(string from, string to, string cc, string bcc, string returnTo, string subject, string body, int format, string[] attachmentNames, byte[][] attachements, string serverName, string port, string userName, string password, string useSSL) Send email with server parameters override.
void	**setCurrentLocale**(string locale) Sets the current locale for the Smartlet
void	**setCurrentPage**(ISmartletPage page)

	Sets the current page.
`void`	**`setHistory`**`(ISmartletPage[] pages)` Set the page visit history.
`ISmartlet`	**`switchSmartlet`**`(ISmartlet anotherSmartlet)` Switches to another `Smartlet`.
`ISmartlet`	**`switchSmartlet`**`(string smartletCode)` Switches to another Smartlet by the given Smartlet code.
`void`	**`triggerEvent`**`(int eventType)` Triggers a specific Smartlet event.

Methods inherited from interface com.alphinat.sg5.ISmartletElement
`accept, data, data, getDataNames, getTypeConst`

Method Detail

getId

`string` **`getId`**`()`

Obtains the unique internal identifier of the Smartlet.

Specified by:

`getId` in interface `ISmartletElement`

Returns:

smartlet ID

---

getName

`string` **`getName`**`()`

Gets the Smartlet name as defined on the properties page of the Smartlet.

Returns:

smartlet name

---

getCode

```
string getCode()
```

Gets the Smartlet code as defined on the properties page of the Smartlet.

Returns:

smartlet code

---

getSubject

```
string getSubject()
```

Gets the Smartlet subject as defined on the properties page of the Smartlet.

Returns:

smartlet subject

---

getKeywords

```
string getKeywords()
```

Gets the Smartlet keywords as defined on the properties page of the Smartlet.

Returns:

smartlet keywords

---

getTheme

```
string getTheme()
```

Gets the theme name of Smartlet as defined on the properties page of the Smartlet.

Returns:

smartlet theme

---

findPageByName

`ISmartletPage` **`findPageByName`**`(string name)`

Finds page by name.

The following example shows how to go to a specific page by its name,

```
ISmartlet smartlet = context.getSmartlet();
ISmartletPage newPage = smartlet.findPageByName(pageName);
smartlet.setCurrentPage(newPage);
```

Parameters:

`name` - Page name.

Returns:

`ISmartletPage`

---

findPageById

`ISmartletPage` **`findPageById`**`(string id)`

Finds a page by id

Parameters:

`id` -

Returns:

`page`

---

findPageByState

`ISmartletPage` **`findPageByState`**`(string state)`

Finds a page by state string.

Parameters:

`state` - - State string, for example: state1, state2...

Returns:

`page`

---

findFieldByName

`ISmartletField` **findFieldByName**(string name)

Finds the first matching field by name.

The following example shows how to reset the value of a field,

```
ISmartlet smartlet = context.getSmartlet();
ISmartletField field = smartlet.findFieldByName(fieldName);
field.setValue("");
```

Parameters:

`name` - - Field name.

Returns:

`smartlet field`

---

findFieldById

`ISmartletField` **findFieldById**(string id)

Finds the first matching field by id

Parameters:

`id` -

Returns:

`smartlet field`

---

getPages

`ISmartletPage[]` **getPages**()

Gets the `pages` of a Smartlet as an Array.

The following example recalculates each page of a Smartlet,

```
ISmartlet smartlet = context.getSmartlet();
ISmartletPage[] pages = smartlet.getPages();
for(int i=0;i<pages.Length;i++)
{
   ISmartletPage page = pages[i];
```

```
    page.calculate();
  }
```

Returns:

pages of smartlet

---

getCurrentPage

`ISmartletPage getCurrentPage()`

Gets the current page.

The following example shows how to trigger the "next" page button of the current page,

```
ISmartlet smartlet = context.getSmartlet();
ISmartletPage page = smartlet.getCurrentPage();
page.navNext();
```

Returns:

current page

---

setCurrentPage

`void setCurrentPage(ISmartletPage page)`

Sets the current page.

Parameters:

`page` - - Smartlet page

---

getHistory

`ISmartletPage[] getHistory()`

Gets the history pages navigated by the user.

Returns:

navigate history

---

setHistory

```
void setHistory(ISmartletPage[] pages)
```

Set the page visit history.

Parameters:

`pages` - - Array of `Smartlet pages`

---

addPageToHistory

```
ISmartlet addPageToHistory(ISmartletPage page)
```

Adds given page to the history.

Parameters:

`page` -

Returns:

this

Since:

5.8.0

---

generateFile

```
byte[] generateFile(int type,
                    string name)
```

Generates a XML or pdf file uploaded to the Smartlet.

Parameters:

`type` - - File type, See `Constants.FileType`

`name` - - File name.

Returns:

generated file bytes.

---

getPDFDictionarypingData

```
Object getPDFDictionarypingData(string pdfFileName)
```

Gets the PDF mapping data.

Parameters:

`pdfFileName` - - PDF file name.

Returns:

Dictionary of pdf field name, value pair

---

generatePDFWithDictionarypingData

```
byte[] generatePDFWithDictionarypingData(string pdfFile,
                                         Object dataDictionary,
                                         bool flatten,
                                         bool readonly)
```

Generates a pdf file with the provided pdf mapping data. The style attribute of each PDF field is ignored.

The following example provides two mappings, for two PDF fields named "fname" and "lname", to be used in generating a PDF file called "reg.pdf",

```
ISmartlet smartlet = context.getSmartlet();
Dictionary map = new Dictionary();
map.Add("fname", "John");
map.Add("lname", "Doe");
byte[] binData = smartlet.generatePDFWithDictionarypingData("reg.pdf", map, true, true);
```

Parameters:

`pdfFile` - - Name of PDF file uploaded in SGD.
`dataDictionary` - - Dictionary of pdf field name - value pair.
`flatten` - - PDF is flattened. This means the resulting PDF is not editable, and the fields appear to be part of the form as static text.
`readonly` - - PDF is readonly. This means the resulting PDF is not editable, but the fields appear inside of their original input boxes as readonly.

Returns:

generated PDF file bytes.

---

getServices

```
ISmartletService[] getServices()
```

Gets the services of the Smartlet.

The following example shows how to iterate through the services, perform a call to a specific service, and get the result,

```
ISmartlet smartlet = context.getSmartlet();
ISmartletService[] services = smartlet.getServices();
ISmartletService service = null;
for(int i=0;i<services.Length;i++)
{
   service = services[i];
   if ("CurrencyLookupWS".Equals(service.getName()))
   {
      if (service.call())
      {
         String xmlResponse = (String)service.getResult();
      }
      break;
   }
}
```

Returns:

all services

---

findServiceByName

`ISmartletService` **`findServiceByName`**`(string name)`

Finds the service by given name.

Parameters:

`name` - - service name

Returns:

- first matching service

Since:

5.4.0

---

getCurrentSmartlet

`ISmartlet` **`getCurrentSmartlet`**`()`

Gets the current `Smartlet`.

Returns:

current `Smartlet`

---

switchSmartlet

`ISmartlet` **`switchSmartlet`**`(ISmartlet anotherSmartlet)`

Switches to another `Smartlet`.

Parameters:

`anotherSmartlet` - - The `Smartlet` the user wants to switch to.

Returns:

`Smartlet` after switch.

---

switchSmartlet

`ISmartlet` **`switchSmartlet`**`(string smartletCode)`

Switches to another Smartlet by the given Smartlet code.

Parameters:

`smartletCode` - - The Smartlet code the user wants to switch to.

Returns:

`Smartlet` after switch.

---

triggerEvent

`void` **`triggerEvent`**`(int eventType)`

Triggers a specific Smartlet event.

The following example demonstrates how to trigger an event when entering a page.

```
ISmartlet smartlet = context.getSmartlet();
smartlet.triggerEvent();
```

These events are defined in the Actions tab of the Smartlet.

Parameters:

`eventType` - - Event type, see `Constants.SmartletEvent`

---

isSubSmartlet

`bool **isSubSmartlet**()`

Check if we are inside a subSmartlet.

Returns:
true if is inside subSmartlet.

---

getParentSubSmartletField

`ISmartletField **getParentSubSmartletField**()`

If the current Smartlet is a subSmartlet, gets the parent subSmartlet field.

The following example demonstrates how to obtain a field value, whose name is "login", from the parent Smartlet. It is assumed that this function is being called while being inside a subSmartlet,

```
ISmartlet subsmartlet = context.getSmartlet();
ISmartletField targetfield = subsmartlet.findFieldByName(fieldName);
ISmartlet parentSmartlet = subsmartlet.getParentSubSmartletField().getSmartlet();
string userId = parentSmartlet.findFieldByName("login").getStringValue();
```

Returns:
parent `subSmartlet field` if inside subSmartlet

---

getGlobalNavButtons

`ISmartletField[] **getGlobalNavButtons**()`

Gets the global navigation buttons.

For a given Smartlet, there is one global navigation button for each page. Triggering a global navigation button allows to go to that page directly.

Returns:
array of `global navigation buttons`.

---

evalBSH

```
Object evalBSH(string bsh)
```

Evaluate Beanshell scripts.

Beanshell is a java source interpreter, and is supported in many areas of SmartGuide (field values, advanced page validations, etc). It can also be executed directly from an extension function.

However, unlike the Beanshell expression written in the Designer, the expressions here should not include the ${ or }$ symbols.

The following is an example of Beanshell script that can be passed as an argument,

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;
zipcode = field("zip").value;
REGEX = "[0-9]{5}";
pSep = Pattern.compile(REGEX);
mSep = pSep.matcher(zipcode);
if (mSep.matches())
{
   page = context5.smartlet.findPageByName("US_Address");
   context5.smartlet.setCurrentPage(page);
}
else
{
   page = context5.smartlet.findPageByName("Error_Page");
   context5.smartlet.setCurrentPage(page);
}
```

This script gets a field value, matches it against a pattern, and then redirects the user to a different page depending on the outcome of the match. Refer to the Beanshell section of the developer's guide for more information on the Beanshell methods available inside SmartGuide.

Parameters:

`bsh` -

Returns:

evaluation result.

createField

```
ISmartletField createField(string fieldName,
                           int fieldType)
```

Create a dynamic field, a unique ID will be assigned to the new field.

Parameters:

`fieldName` - - name of field to create.

`fieldType` - - type of field to create. See `Constants.FileType`

Returns:

Since:

5.4.0

---

createField

```
ISmartletField createField(string newFieldId,
                           string fieldName,
                           int fieldType)
```

Create a dynamic field.

Parameters:

`newFieldId` - - ID of field to create. If this parameter is null or empty, a unique ID will be assigned to the new field;

`fieldName` - - name of field to create.

`fieldType` - - type of field to create. See `Constants.FileType`

Returns:

Since:

5.4.0

---

createField

```
ISmartletField createField(ISmartletField anotherField)
```

Create a dynamic field from existing field, a unique ID will be assigned to the new field.

Parameters:

`anotherField` - - template field to be created from.

Returns:

---

createField

```
ISmartletField createField(string newFieldId,
                           ISmartletField anotherField)
```

Create a dynamic field from existing field.

Parameters:

`newFieldId` - - ID of field to create. If this parameter is null or empty, a unique ID will be assigned to the new field;

`anotherField` - - template field to be created from.

Returns:

---

getProgress

```
string getProgress()
```

Returns the current percentage complete, from "0" to "100"

Returns:

- string from "0" to "100"

Since:

5.4.0

---

getSubSmartletReturnButton

```
ISmartletField getSubSmartletReturnButton()
```

Gets the button to return from subsmartlet. Returns null if not under sub smartlet.

Returns:

- button

Since:

5.4.0

---

getSubSmartletCancelButton

```
ISmartletField getSubSmartletCancelButton()
```

Gets the button to return from sub smartlet without save. Returns null if not under sub smartlet.

Returns:

- button

Since:

5.4.0

---

getDomain

```
string getDomain()
```

Gets current domain name

Returns:

- domain name

Since:

5.4.0

---

getWorkspace

```
string getWorkspace()
```

Gets current workspace name

Returns:

- workspace name

Since:

5.4.0

---

getCurrentLocale

```
string getCurrentLocale()
```

Gets the current locale for the Smartlet

Returns:

- locale (e.g. en-US)

Since:

5.6.0

---

getCurrentLocaleDescription

```
string getCurrentLocaleDescription()
```

Gets the current language for the Smartlet

Returns:

- language (e.g. English)

Since:

5.6.0

---

setCurrentLocale

```
void setCurrentLocale(string locale)
```

Sets the current locale for the Smartlet

Parameters:

`locale` - - locale to set (e.g. en-US).

Since:

5.6.0

---

getLocales

```
string[] getLocales()
```

Gets the array of locales supported by the Smartlet

Returns:

- locales array

Since:

5.6.0

---

getLocalesDescription

```
string[] getLocalesDescription()
```

Gets the array of locales description (languages) supported by the Smartlet

Returns:

- locales description array

Since:

5.6.0

---

getLocalizedResource

```
string getLocalizedResource(string key)
```

Gets value corresponding to a custom key for the resources

Parameters:

`key` - - key as specified in the resources (e.g. theme.text.progress).

Returns:

- value corresponding to the translated text for the key

Since:

5.6.0

---

addLocalizedResource

```
void addLocalizedResource(string locale,
                          string key,
                          string value)
```

Add a key/value pair to the translation resources

Parameters:

`locale` - - locale to set (e.g. en-US).

`key` - - key for the resources (e.g. theme.text.progress).

`value` - - value corresponding to the specified key. If null, this will remove the key from the resources.

Since:

5.6.0

---

calculate

```
void calculate()
```

Recalculates the page.

Since:

5.8.0

---

clear

```
void clear()
```

Calls clear on every page of this smartlet, recursively.

Since:

5.8.0

---

gotoPage

```
ISmartletPage gotoPage(long pageId)
                        throws java.lang.Exception
```

Navigate to page with given page id and add current page to history. Will throw if page not found. This overload will call services on page entry/exit and add current page to history

Parameters:

`pageId` -

Returns:

this

Throws:

`java.lang.Exception`

Since:

6.0.0

---

gotoPage

```
ISmartletPage gotoPage(string pageName)
                        throws java.lang.Exception
```

Navigate to page with given page name and add current page to history. Will throw if page not found. Use tryGotoPage if unsure. This overload will call services on page entry/exit and add current page to history

Parameters:

`pageName` -

Returns:

this

Throws:

`java.lang.Exception`

Since:

5.8.0

---

gotoPage

```
ISmartletPage gotoPage(string pageName,
                        bool addCurrentPageToHistory)
                        throws java.lang.Exception
```

Navigate to page with given page name and add current page to history. Will throw if page not found. Use tryGotoPage if unsure. This overload will call services on page entry/exit

Parameters:

`pageName` -

`addCurrentPageToHistory` -

Returns:

this

Throws:

`java.lang.Exception`

Since:

5.8.0

---

gotoPage

```
ISmartletPage gotoPage(string pageName,
                       bool addCurrentPageToHistory,
                       bool callServicesOnPageExit,
                       bool callServicesOnPageEntry)
                       throws java.lang.Exception
```

Navigate to page with given page name and add current page to history. Will throw if page not found. Use tryGotoPage if unsure.

Parameters:

`pageName` -

`addCurrentPageToHistory` -

`callServicesOnPageExit` -

`callServicesOnPageEntry` -

Returns:

this

Throws:

`java.lang.Exception`

Since:

5.8.0

---

gotoPage

```
ISmartletPage gotoPage(string pageName,
                       bool preserveHistory,
                       bool callServicesOnPageExit,
                       bool callServicesOnPageEntry,
                       string[] pageNamesToAddToHistory)
                       throws java.lang.Exception
```

Navigate to page with given page name. If preserveHistory is true, the existing history will be preserved and the current page will be added to it. If it is false, then the existing history will be cleared and the current page will not be added to the history. If pageNamesToAddToHistory is specified, and preserveHistory is false, then all pages specified in the array will be put in the history. Will throw if page not found. Use tryGotoPage if unsure.

Parameters:

`pageName` -

`preserveHistory` -

`callServicesOnPageExit` -

`callServicesOnPageEntry` -

`pageNamesToAddToHistory` -

Returns:

this

Throws:

`java.lang.Exception`

Since:

6.0.0

---

gotoSmartlet

```
ISmartlet gotoSmartlet(string smartletCode,
                       string destinationPageId,
                       bool preserveHistoryAndData,
                       bool callServicesOnPageExit)
                       throws java.lang.Exception
```

Navigate to Smartlet with given code. If preserveHistoryAndData is true, the current Smartlet session object will be preserved in session and restored if returning to the current Smartlet. If it is false, then when returning to the current Smartlet everything will be reinitialized. If a destination page id is provided, this will be set as the current page when entering the destination Smartlet. If callServicesOnPageExit is true, the services on page exit will be called. Will throw if page not found. Use tryGotoSmartlet if unsure.

Parameters:

`smartletCode` -

`destinationPageId` -

`preserveHistoryAndData` -

`callServicesOnPageExit` -

Returns:

destination Smartlet

Throws:

`java.lang.Exception`

Since:

6.0.0

---

hasPage

```
bool hasPage(string pageName)
```

Check if this smartlet

Parameters:

`pageName` -

Returns:

---

addActionError

```
void addActionError(Object sourceObject,
                    string error,
                    string callStack)
```

Adds an action error element composed of a source object, the error message and a stack trace.

Parameters:

`sourceObject` - - normally the class where the error is being logged.

`error` - - error message describing the error.

`callStack` - - the stack trace where the error occured.

Since:

6.0.0

---

addActionError

```
void addActionError(ISmartletActionError actionError)
```

Adds an action error object.

Parameters:

`actionError` - - ISmartletActionError object.

Since:

6.5.0

---

clearActionErrors

```
void clearActionErrors()
```

Clears all action error of this smartlet.

Since:

6.0.0

---

getActionErrors

```
ISmartletActionError[] getActionErrors()
```

Gets an array of `errors` for the current Smartlet.

Returns:

`errors` of smartlet

---

sendMail

```
void sendMail(string from,
              string to,
              string cc,
              string bcc,
              string returnTo,
              string subject,
              string body,
              int format,
              string[] attachmentNames,
              byte[][] attachements)
              throws java.lang.Exception
```

Send email.

Parameters:

`from` - - The sender's email address

`to` - - The recipient's email address

`cc` - - Addresses of recipients who will also receive copies

`bcc` - - Hide the address list from all the recipients

`returnTo` - - The recipient where non-delivery receipts are to be sent.

`subject` - - Subject of the message as specified by the sender.

`body` - - Body of message.

`format` - - `email formats`

`attachmentNames` - - array of attachement names, array size should be the same as attachements.

`attachements` - - bytes of attachements. Array size should be the same as parameter attachmentNames. Otherwise throws Esxception.

Throws:

`java.lang.Exception`

Since:

6.5.0

sendMail

```
void sendMail(string from,
              string to,
              string cc,
              string bcc,
              string returnTo,
              string subject,
              string body,
              int format,
              string[] attachmentNames,
              byte[][] attachements,
              string serverName,
              string port,
              string userName,
              string password,
              string useSSL)
              throws java.lang.Exception
```

Send email with server parameters override.

Parameters:

`from` - - The sender's email address

`to` - - The recipient's email address

`cc` - - Addresses of recipients who will also receive copies

`bcc` - - Hide the address list from all the recipients

`returnTo` - - The recipient where non-delivery receipts are to be sent.

`subject` - - Subject of the message as specified by the sender.

`body` - - Body of message.

`format` - - `email formats`

`attachmentNames` - - array of attachement names, array size should be the same as attachements.

`attachements` - - bytes of attachements. Array size should be the same as parameter attachmentNames. Otherwise throws Esxception.

`serverName` - - address of mail server.

`port` - - port of mail server.

`userName` - - username to connect to server.

`password` - - password to connect to server.

`useSSL` - - can be true of false to specify whether SSL should be used.

Throws:

`java.lang.Exception`

Since:

6.5.0

com.alphinat.sg5

Interface ISmartletPage

All Superinterfaces:

ISmartletElement

`public interface` **`ISmartletPage`**
extends ISmartletElement

`ISmartletPage` is an interface representing a Smartlet page along with the operations that may be performed on that page.

Method Summary

`void`	**`calculate`**`()` Recalculates the page.
`void`	**`clear`**`()` Calls clear on every field of this page, recursively.
`ISmartletField[]`	**`findAllFields`**`()` Returns all fields of the page.
`ISmartletField[]`	**`findErrorFields`**`()` Returns fields with error.
`ISmartletField`	**`findFieldById`**`(string id)` Finds the first matching field by id
`ISmartletField`	**`findFieldByName`**`(string name)` Finds the first matching field by name.
`ISmartletField[]`	**`findFieldsByRegex`**`(string regularExpression)` Finds fields by regulation expression of the page.
`ISmartletField[]`	**`findFieldsByScript`**`(string script)` Finds fields by matching script of the page.

ISmartletField[]	**findFieldsByTypes**(int[] types) Finds fields by types of the page.
string	**getCSSClass**() Gets the css class.
string	**getCSSStyle**() Gets the css style.
int[]	**getErrorCodes**() Gets the page level validation error codes.
string[]	**getErrorMessages**() Gets the page level validation error messages.
ISmartletField[]	**getFields**() Gets the page fields.
string	**getId**() Obtains the unique internal identifier of the page.
ISmartletField	**getModifyPageButton**() Gets the "modify" button under the summary section for this page.
string	**getName**() Obtains the user-defined name of the page.
ISmartletField	**getNavNextButton**() Gets the next page button
ISmartletField	**getNavPreviousButton**() Gets the previous page button
ISmartletField	**getNavSummaryButton**() Gets the navigate to summary page button
ISmartletField[]	**getShownFields**() Gets the top level of page available fields.
ISmartlet	**getSmartlet**() Gets the Smartlet this page belongs to.
string	**getState**()

	Gets the page state string.
`string`	**`getTemplate`**`()` Gets the page template.
`string`	**`getTitle`**`()` Obtains the user-defined title of the page.
`bool`	**`isShownInSummarySection`**`()` Returns true if the page is shown under the summary section
`bool`	**`isValid`**`()` Is the page valid or not.
`void`	**`navNext`**`()` Performs navigation to the next page.
`void`	**`navPrevious`**`()` Performs navigation to the previous page.
`void`	**`resetValidationResult`**`()` Cleans the validation results and error messages.
`void`	**`setCSSClass`**`(string str)` Sets the css class.
`void`	**`setCSSStyle`**`(string str)` Sets the css style.
`void`	**`setTitle`**`(string title)` Specifies the page title.
`void`	**`setTitle`**`(string title, string locale)` Specifies the page title for a specific locale.
`bool`	**`validate`**`()` Validates the page.

Methods inherited from interface com.alphinat.sg5.ISmartletElement

`accept, data, data, getDataNames, getTypeConst`

Method Detail

getId

`string getId()`

Obtains the unique internal identifier of the page.

Specified by:

`getId` in interface `ISmartletElement`

Returns:

the `string` representing the unique internal identifier of the page.

---

getName

`string getName()`

Obtains the user-defined name of the page.

The name of the page is defined on the Properties tab of the Smartlet page.

Returns:

the `string` representing the user-defined name of the page.

---

getTitle

`string getTitle()`

Obtains the user-defined title of the page.

The title of the page is defined on the Properties tab of the Smartlet page.

Returns:

the `string` representing the user-defined title of the page.

---

setTitle

`void setTitle(string title)`

Specifies the page title.

Allows to specify a title for the page.

Parameters:

`title` - - title to assign to the page

Since:

6.5.0

---

setTitle

```
void setTitle(string title,
              string locale)
```

Specifies the page title for a specific locale.

Allows to specify a title for the page.

Parameters:

`title` - - title to assign to the page

`locale` - - locale for which the title will be set

Since:

6.5.0

---

getCSSClass

```
string getCSSClass()
```

Gets the css class.

The css class is defined on the Appearance tab when editing a page in the Designer.

Returns:

css class of page.

---

setCSSClass

```
void setCSSClass(string str)
```

Sets the css class.

Parameters:

`str` - - new css class.

Since:
7.0.0

getCSSStyle

```
string getCSSStyle()
```

Gets the css style.

The css style is defined on the Appearance tab when editing a page in the Designer.

Returns:
css style of page.

setCSSStyle

```
void setCSSStyle(string str)
```

Sets the css style.

Parameters:
`str` - - new css style.

Since:
7.0.0

getState

```
string getState()
```

Gets the page state string.

Returns:
state of page. Ex. state1, state2...

getTemplate

```
string getTemplate()
```

Gets the page template.

The page template is defined on the Appearance tab when editing a Smartlet page.

Returns:

page template.

---

getSmartlet

`ISmartlet` **`getSmartlet`**`()`

Gets the Smartlet this page belongs to.

Returns:

`ISmartlet` the page belongs to.

---

getFields

`ISmartletField[]` **`getFields`**`()`

Gets the page fields.

Returns:

array of `ISmartletField` on the page

---

getNavNextButton

`ISmartletField` **`getNavNextButton`**`()`

Gets the next page button

Returns:

next button.

---

getNavPreviousButton

`ISmartletField` **`getNavPreviousButton`**`()`

Gets the previous page button

Returns:

previous button.

---

getNavSummaryButton

`ISmartletField getNavSummaryButton()`

Gets the navigate to summary page button

Returns:
button to navigate to summary page.

---

getModifyPageButton

`ISmartletField getModifyPageButton()`

Gets the "modify" button under the summary section for this page.

Returns:
button to navigate to page.

---

validate

`bool validate()`

Validates the page.

Returns:
true if page is valid.

---

isValid

`bool isValid()`

Is the page valid or not. Page is valid if there is no error field and no page level validation error.

Returns:
true if no errors on the page.

---

calculate

```
void calculate()
```

Recalculates the page.

navNext

```
void navNext()
```

Performs navigation to the next page.

This is functionally equivalent to the following call.

```
ISmartletPage page = context.getSmartlet().getCurrentPage()();
page.getNavNextButton().triggerEvent();
```

navPrevious

```
void navPrevious()
```

Performs navigation to the previous page.

This is functionally equivalent to the following call.

```
ISmartletPage page = context.getSmartlet().getCurrentPage()();
page.getNavPreviousButton().triggerEvent();
```

getErrorMessages

```
string[] getErrorMessages()
```

Gets the page level validation error messages.

Returns:

error messages

getErrorCodes

`int[] getErrorCodes()`

Gets the page level validation error codes.

Returns:
error codes
See Also:
`{@link Constants.ErrorCode}`

---

findErrorFields

`ISmartletField[] findErrorFields()`

Returns fields with error. Returns empty array of no error found.
Returns:
`array of Smartlet fields`
Since:
5.4.0

---

findFieldsByTypes

`ISmartletField[] findFieldsByTypes(int[] types)`

Finds fields by types of the page.
Parameters:
`types` - - field types to find
Returns:
`array of Smartlet fields`
Since:
5.4.0

---

findFieldByName

`ISmartletField findFieldByName(string name)`

Finds the first matching field by name.

The following example shows how to reset the value of a field,

```
ISmartletPage page = context.getSmartlet().getCurrentPage();
ISmartletField field = page.findFieldByName(fieldName);
field.setValue("");
```

Parameters:

`name` - - Field name.

Returns:

`smartlet field`

Since:

7.0.0

---

findFieldById

`ISmartletField` **`findFieldById`**`(string id)`

Finds the first matching field by id

Parameters:

`id` -

Returns:

`smartlet field`

Since:

7.0.0

---

findFieldsByRegex

`ISmartletField[]` **`findFieldsByRegex`**`(string regularExpression)`

Finds fields by regulation expression of the page.

Parameters:

`regularExpression` - - regulation expression to match the field name

Returns:

`array of Smartlet fields`

Since:

5.4.0

---

findFieldsByScript

`ISmartletField[] findFieldsByScript(string script)`

Finds fields by matching script of the page.

Parameters:

`script` - - BSH script

Returns:

`array of Smartlet fields`

Since:

5.4.0

---

findAllFields

`ISmartletField[] findAllFields()`

Returns all fields of the page.

Returns:

`array of Smartlet fields`

---

getShownFields

`ISmartletField[] getShownFields()`

Gets the top level of page available fields.

Returns:

array of `ISmartletField` on the page

Since:

5.4.0

---

isShownInSummarySection

`bool isShownInSummarySection()`

Returns true if the page is shown under the summary section

Returns:

- bool

Since:

5.4.0

clear

```
void clear()
```

Calls clear on every field of this page, recursively.

Since:

5.8.0

resetValidationResult

```
void resetValidationResult()
```

Cleans the validation results and error messages.

Since:

7.0.0

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5

Interface ISmartletField

All Superinterfaces:

ISmartletElement

All Known Subinterfaces:

ISmartletDate, ISmartletGroup, ISmartletKnowledge, ISmartletRepeat, ISmartletSelectField, ISmartletSummary, ISmartletUpload, ISubSmartletField

```
public interface ISmartletField
```
extends ISmartletElement

`ISmartletField` is an interface representing a Smartlet field along with the operations that may be performed on that field.

Method Summary

`void`	**`addSourceField`**`(string fieldId)` Add a source field to the current field, for autorefresh purposes.
`void`	**`addTargetField`**`()` Add the current field as a target to itself, for autorefresh purposes.
`void`	**`addTargetField`**`(string fieldId)` Add a target field to the current field, for autorefresh purposes.
`void`	**`addTargetFieldByName`**`(string fieldName)` Add a target field to the current field, for autorefresh purposes.
`void`	**`appendAfter`**`(ISmartletField field)` Appends the field after given field.
`void`	**`appendBefore`**`(ISmartletField field)` Appends the field before given field.
`void`	**`appendTo`**`(ISmartletField parent, int position)` Appends field under the parent field at the specified position .

void	**appendTo**(ISmartletPage page, int position) Appends field to page at given position.
void	**applyDefinition**() Changes field definition.
void	**calculate**() Recalculates the field value.
bool	**calculateAvailability**() Calculates and returns the field availability.
void	**clear**() Clears the fields.
void	**detach**() Detaches a field.
string	**getChoiceLayout**() Gets the layout of choices for select type field.
string	**getCSSClass**() Gets the css class.
string	**getCSSHeight**() Gets the css height.
string	**getCSSStyle**() Gets the css style.
string	**getCSSWidth**() Gets the css width.
int[]	**getErrorCodes**() Gets the validation error codes of the field.
string[]	**getErrorMessages**() Gets the validation error messages of the field.
ISmartletField[]	**getEventSource**() Returns an array of ISmartletField which have an impact, through validation rules, dynamic values or visibility conditions on the current field.
ISmartletField[]	**getEventTarget**() Returns an array of ISmartletField which are impacted, through validation rules, dynamic values or visibility conditions by the current field.

string	**getFormat**() Obtains the name of the format used during field validation.
string	**getHelp**() Gets the help text.
string	**getHelpId**() Gets the help id used to render help link.
string	**getHtmlName**() Gets the html name.
string	**getId**() Obtains the unique internal identifier of the field.
string	**getLabel**() Obtains the label of the field.
string	**getLayoutAttribute**(string deviceName, string attributeName) Get layout attribute string by device and attribute name.
string	**getLayoutAttributes**() Get layout attributes string.
int	**getMaxLength**() Gets the maximum length.
string	**getMetaData**(string name) Get meta data value by name.
string[]	**getMetaDataNames**() Get all meta data names.
int	**getMinLength**() Gets the minimum length.
string	**getName**() Obtains the user-defined name of the field.
ISmartletField	**getNext**() Gets the next sibling field.
ISmartletPage	**getPage**() Gets the page that this field belongs to.
ISmartletField	

	getParent() Gets the parent field.
string	**getPlacement**() Gets the field placement definition.
string	**getPrefix**() Gets the field prefix text.
ISmartletField	**getPrevious**() Gets the previous sibling field.
int[]	**getRepeatIndex**() Returns an array of integers representing repeatable group indices.
string[]	**getRepeatSelectedStrings**() Returns strings for selected groups in repeated field.
Object[]	**getRepeatSelectedValues**() Returns values for selected groups in repeated field.
string[]	**getRepeatStrings**() Returns strings for repeated field.
Object[]	**getRepeatValues**() Returns values for repeated field.
ISmartlet	**getSmartlet**() Gets the Smartlet that this field belongs to.
string	**getString**() Gets the `string` value of the field.
string	**getSuffix**() Gets the field suffix text.
string	**getTooltip**() Gets the tool tip text.
int	**getTypeConst**() Gets the type constant.
string	**getTypeDetail**() Gets the detailed type information of Smartlet field.
Object	**getValue**() Obtains the value of the field.

`string`	**`getValueParseScript`**`()` Gets script to parse the value
`bool`	**`isAvailable`**`()` Is the field available.
`bool`	**`isEncrypted`**`()` Gets the "encrypt" flag.
`bool`	**`isHelpLink`**`()` Is help text a link or not.
`bool`	**`isPersistent`**`()` Gets the "persist" flag.
`bool`	**`isReadonly`**`()` Determines whether the field is read-only or not.
`bool`	**`isRequired`**`()` Is the field mandatory or not.
`bool`	**`isRequiredOnSummaryOnly`**`()` Is the field mandatory only on summary or not.
`bool`	**`isShownInSummarySection`**`()` Returns true if the field is shown under the summary section
`bool`	**`isUnderRepeatDefaultGroup`**`()` Returns true if field is under repeat default group or is default group itself.
`bool`	**`isValid`**`()` Is the field valid or not.
`void`	**`removeSourceField`**`(string fieldId)` Removes the specified field from the current list of source fields for the current field, for autorefresh purposes.
`void`	**`removeTargetField`**`()` Removes the current field from itself as a target, for autorefresh purposes.
`void`	**`removeTargetField`**`(string fieldId)` Removes the specified field from the current list of target fields for the current field, for autorefresh purposes.

void	**removeTargetFieldByName**(string fieldName) Removes the specified field from the current list of target fields for the current field, for autorefresh purposes.
void	**resetValidationDefinitions**() Resets the validation definitions.
void	**resetValidationResult**() Cleans the validation results and error messages.
void	**setAvailabilityScript**(string script) Defines script to calculate field availability.
void	**setCalculationScript**(string script, bool alwaysCalc) Defines field calculation script.
void	**setChoiceLayout**(string str) Sets the layout of choices for select type field.
void	**setCSSClass**(string str) Sets the css class.
void	**setCSSHeight**(string str) Sets the css height.
void	**setCSSStyle**(string str) Sets the css style.
void	**setCSSWidth**(string str) Sets the css width.
void	**setEncrypted**(bool encrypt) Sets the "encrypt" flag.
void	**setError**(string errorMessage) Manually sets field error message.
void	**setFormat**(string format, string errorMessage) Set field format validation.
void	**setHelp**(string help) Sets the help text of field.
void	**setLabel**(string label) Modifies the label of the field.

`void`	**setLayoutAttributes**(string layoutString) Set layout attributes string.
`void`	**setMetaData**(string name, string value) Set meta data.
`void`	**setPersistent**(bool persitent) Sets the "persist" flag
`void`	**setPlacement**(string str) Sets the field placement definition.
`void`	**setPosition**(int rowPos, int colPos) Set field position to new row, column of current layout (row and column starts with 0).
`void`	**setPrefix**(string str) Sets the field prefix text.
`ISmartletField`	**setReadonly**(bool readonly) Sets the "readonly" flag
`ISmartletField`	**setReadonly**(bool readonly, java.util.Collection<string> exceptions) Sets the "readonly" flag
`void`	**setRepeatStrings**(string[] strings) Sets the repeated field with strings.
`void`	**setRepeatValues**(Object[] values) Sets the repeated field with values.
`void`	**setString**(string val) Sets the `string` value of the field.
`void`	**setSuffix**(string str) Sets the field suffix text.
`void`	**setTooltip**(string tooltip) Sets the tooltip text of field.
`void`	**setValidateOnSummaryOnly**(bool bool) If sets to true, the validation will happen only on summary section.
`void`	**setValidationRequire**(string isRequiredScript,

	`string errorMessage)` Defines required validation.
`void`	**`setValidationScript`**`(string validationiScript)` Defines script validation
`void`	**`setValue`**`(Object value)` Set the field value object.
`void`	**`setValueParseScript`**`(string valueParseScript)` Defines script to parse value.
`void`	**`triggerEvent`**`()` Triggers the event associated to a field.
`bool`	**`validate`**`()` Revalidates the field.

Methods inherited from interface com.alphinat.sg5.ISmartletElement

`accept`, `data`, `data`, `getDataNames`

Method Detail

getId

```
string getId()
```

Obtains the unique internal identifier of the field.

Specified by:

`getId` in interface `ISmartletElement`

Returns:

the `string` representing the unique internal identifier of the field.

getTypeConst

```
int getTypeConst()
```

Gets the type constant. See also `Constants.ElementType`.

Specified by:

`getTypeConst` in interface `ISmartletElement`

Returns:

the `int` representing field type.

---

getTypeDetail

```
string getTypeDetail()
```

Gets the detailed type information of Smartlet field. See also `Constants.ElementType`.

Type detail for buttons:

```
  BUTTON_SUBSMARTLET_ENTER
  BUTTON_SUBSMARTLET_RETURN_SAVE
  BUTTON_SUBSMARTLET_WITHOUT_SAVE
  BUTTON_NEXT_PAGE
  BUTTON_PREVIOUS_PAGE
  BUTTON_REPEAT_INSERT
  BUTTON_REPEAT_DELETE
  BUTTON_GOTO_SUMMARY
  BUTTON_MODIFY_PAGE
  BUTTON_REFRESH_PAGE
  BUTTON_GLOBAL_NAVIGATION
```

Returns:

the `string` representing field type details.

---

getName

```
string getName()
```

Obtains the user-defined name of the field.

Returns:

the `string` representing the user-defined name of the field.

---

getLabel

```
string getLabel()
```

Obtains the label of the field.

Returns:

the `string` representing the label of the field.

---

setLabel

```
void setLabel(string label)
```

Modifies the label of the field. The modification only affects the session copy of the field.

Parameters:

`label` - - the `string` representing the new field label.

---

getValue

```
Object getValue()
```

Obtains the value of the field.

within JAVA extension function or BeanShell:

Return `string` for text field of select field.
Return `Number` for number field.
Return `Date` for date field.
Return `ISmartlet` for sub smartlet field.
Return `byte[]` for upload field.

within DOTNET extension function:

Return `string` for text field of select field.
Return `long` or `double` for number field.
Return `DateTime` for date field.
Return `ISmartlet` for sub smartlet field.
Return `byte[]` for upload field.

Returns:

the `Object` representing the value of the field.

---

setValue

```
void setValue(Object value)
```

Set the field value object.

within JAVA extension function or BeanShell:

`string` for text field of select field.
`Number` or `string` for number field.
`Date` or `string` (yyyy-mm-dd format) for date field.
`ISmartlet` for subsmartlet field.
`byte[]` for upload field.

within DOTNET extension function:

`string` for text field of select field.
`long`, `double` etc (DOTNET primitive number type) or `string` for number field.
`DateTime` or `string` (yyyy-mm-dd format) for date field.
`ISmartlet` for subsmartlet field.
`byte[]` for upload field.

Parameters:

`value` - - field value

---

getString

```
string getString()
```

Gets the `string` value of the field.

Returns:

string value.

---

setString

```
void setString(string val)
```

Sets the `string` value of the field.

Parameters:

`val` -

---

getFormat

```
string getFormat()
```

Obtains the name of the format used during field validation. If no format has been associated to the field, `null` is returned. If conditional formatting is in effect where multiple formats have been applied to the field, only the first format name is returned.

The following example gets the format of a date field which has been defined as yyyy-mm-dd in the Designer for that field.

```
ISmartletField datefield = context.getSmartlet().findFieldByName("birth date");
string format = datefield.getFormat();
// ...will return yyyy-mm-dd
```

Returns:

the `string` representing the name of the format used during field validation.

---

setFormat

```
void setFormat(string format,
               string errorMessage)
```

Set field format validation. If format is null or empty, the original format definition will be applied.

Format is defined by following pattern strings:

◊ Date format for date field, e.g. yyyy-mm-dd or aaaa-mm-jj (French format). The following pattern letters are defined:

1. y/a - Year

2. m - Month digit in year, from 1 to 12
3. d/j - Day in month
4. month/mois - Month text

◊ number format for number format, e.g. ###,###.00. The following pattern letters are defined:

1. # - Digit, zero shows as absent
2. 0 - Digit
3. . - Decimal separator or monetary decimal separator
4. , - Grouping separator

◊ For other type of fields, the format string is the name of format defined in SmartGuideDesigner.

Parameters:

`format` - - pattern string

`errorMessage` -

Since:

5.4.0

---

getHelp

```
string getHelp()
```

Gets the help text.

The help text is defined on the Contextual help tab in the Detailed help field in the Designer.

Returns:

help text of field.

---

setHelp

```
void setHelp(string help)
```

Sets the help text of field.

Parameters:

`help` - - help text.

---

getHelpId

```
string getHelpId()
```

Gets the help id used to render help link.

Returns:
help id to render link to help.

---

isHelpLink

```
bool isHelpLink()
```

Is help text a link or not.

This returns true or false depending on whether the Hyperlink checkbox has been checked for that field on the Contextual help tab in the Designer.

Returns:
true if help text is a link.

---

getHtmlName

```
string getHtmlName()
```

Gets the html name.

This name corresponds to the assigned name for the field in the generated html for the page and usually takes the form "d_xyz" for fields and "t_xyz" for buttons, where xyz is the ID of the field.

Returns:
html name of field.

---

getTooltip

```
string getTooltip()
```

Gets the tool tip text.

The tooltip text is defined on the Contextual help tab in the Tooltip field in the Designer.

Returns:

field tooltip text.

---

setTooltip

```
void setTooltip(string tooltip)
```

Sets the tooltip text of field.

Parameters:

`tooltip` -

---

getCSSClass

```
string getCSSClass()
```

Gets the css class.

The css class is defined on the Appearance tab when editing a field in the Designer.

Returns:

css class of field.

---

setCSSClass

```
void setCSSClass(string str)
```

Sets the css class.

Parameters:

`str` - - new css class. Reset to original definition if null.

Since:

5.4.0

---

getCSSStyle

```
string getCSSStyle()
```

Gets the css style.

The css style is defined on the Appearance tab when editing a field in the Designer.

Returns:
css style of field.

---

setCSSStyle

```
void setCSSStyle(string str)
```

Sets the css style.

Parameters:
`str` - - new css style. Reset to original definition if null.

Since:
5.4.0

---

getCSSWidth

```
string getCSSWidth()
```

Gets the css width.

The css width is defined on the Appearance tab when editing a field in the Designer.

Returns:
css width.

---

setCSSWidth

```
void setCSSWidth(string str)
```

Sets the css width.

Parameters:
`str` - - new css width. Reset to original definition if null.

Since:
5.4.0

---

getCSSHeight

```
string getCSSHeight()
```

Gets the css height.

The css height is defined on the Appearance tab when editing a field in the Designer.

Returns:

css height

---

setCSSHeight

```
void setCSSHeight(string str)
```

Sets the css height.

Parameters:

`str` - - new css height. Reset to original definition if null.

Since:

5.4.0

---

getPlacement

```
string getPlacement()
```

Gets the field placement definition.

The field placement is defined on the Appearance tab when editing a field in the Designer. It is used to determine whether a field should be displayed next or under the previous field.

Returns:

field placement

---

setPlacement

```
void setPlacement(string str)
```

Sets the field placement definition.

Parameters:

`str` - - new field placement. Reset to original definition if null.

Since:

5.4.0

---

getChoiceLayout

```
string getChoiceLayout()
```

Gets the layout of choices for select type field.

The choice layout is defined on the Appearance tab when editing a field in the Designer. It is used to determine, for check box and radio button fields, whether the options should be under each other or one the same line.

Returns:

layout of choices for select type field. The values set by Designer are "default", "vertically" and "horizontally".

---

setChoiceLayout

```
void setChoiceLayout(string str)
```

Sets the layout of choices for select type field.

Parameters:

`str` - - new choice layout. Reset to original definition if null. The values used by Designer are "default", "vertically" and "horizontally".

Since:

5.4.0

---

getPrefix

```
string getPrefix()
```

Gets the field prefix text.

The prefix is defined on the Appearance tab when editing a field in the Designer.

Returns:

field prefix text.

---

setPrefix

```
void setPrefix(string str)
```

Sets the field prefix text.

Parameters:

`str` - - new field prefix text. Reset to original definition if null.

Since:

5.4.0

---

getSuffix

```
string getSuffix()
```

Gets the field suffix text.

The suffix is defined on the Appearance tab when editing a field in the Designer.

Returns:

field suffix text.

---

setSuffix

```
void setSuffix(string str)
```

Sets the field suffix text.

Parameters:

`str` - - new field suffix text. Reset to original definition if null.

Since:

5.4.0

---

getPage

```
ISmartletPage getPage()
```

Gets the page that this field belongs to.

Returns:

`ISmartletPage` where the field is located.

---

getSmartlet

`ISmartlet` **`getSmartlet`**`()`

Gets the Smartlet that this field belongs to.

Returns:
`ISmartlet` where the field is located.

---

getParent

`ISmartletField` **`getParent`**`()`

Gets the parent field.

The new API in version 5 and above gives access to the hierarchical structure of the fields on a page. For example if a group with name "genericInfo" contains a field called "name", the following code example would return the group object given the field name.

```
ISmartlet smartlet = context.getSmartlet();
ISmartletField namefield = smartlet.findFieldByName()("name");
ISmartletGroup parentGroup = namefield.getParent();
```

Returns:
parent field.

---

getNext

`ISmartletField` **`getNext`**`()`

Gets the next sibling field.

Returns:
next field.

---

getPrevious

`ISmartletField` **`getPrevious`**`()`

Gets the previous sibling field.

Returns:

previous field.

---

getRepeatIndex

```
int[] getRepeatIndex()
```

Returns an array of integers representing repeatable group indices. If the field is not under a repeatable group, returns null;

For example suppose one has a repeatable group called "personalInfo". In such a group one can enter information for several persons. One such piece of information could be another repeatable group called "emailAdresses" containing several email addresses for each person. If an event is triggered by one of these email entries, then its indexed position across the two level of groups can be obtained by getRepeatIndex().

Returns:

array of repeat index position.

---

getMaxLength

```
int getMaxLength()
```

Gets the maximum length.

Returns the maximum length of the field as defined on the Appearance tab of the field under the Designer.

Returns:

max length.

---

getMinLength

```
int getMinLength()
```

Gets the minimum length.

Returns the minimum length of the field as defined on the Appearance tab of the field under the Designer.

Returns:

min length

isEncrypted

```
bool isEncrypted()
```

Gets the "encrypt" flag.

This flag can be specified on the Properties tab of the field under the Designer.

Returns:

true if encrypt flag is set.

setEncrypted

```
void setEncrypted(bool encrypt)
```

Sets the "encrypt" flag. It is a flag that can be used inside extension functions. The user needs to implement the encryption.

Parameters:

`encrypt` - - encrypted or not.

isPersistent

```
bool isPersistent()
```

Gets the "persist" flag. It is a flag that can be used inside extension functions. The user needs to implement the persistence.

This flag can be specified on the Properties tab of the field under the Designer.

Returns:

true if persist flag is set.

setPersistent

```
void setPersistent(bool persitent)
```

Sets the "persist" flag

Parameters:

`persitent` - - persistent or not.

---

isReadonly

```
bool isReadonly()
```

Determines whether the field is read-only or not. Modifying read-only fields does not have any effect on the Smartlet.

Returns:

`true` if the field is read-only, `false` otherwise.

---

setReadonly

```
ISmartletField setReadonly(bool readonly)
```

Sets the "readonly" flag

This flag can be specified on the Properties tab of the field under the Designer.

Parameters:

`readonly` - - readonly or not.

Returns:

this

Since:

5.8.0 - this call is now recursive.

---

setReadonly

```
ISmartletField setReadonly(bool readonly,
                           java.util.Collection<string> exceptions)
```

Sets the "readonly" flag

This flag can be specified on the Properties tab of the field under the Designer.

Parameters:

`readonly` - - readonly or not.

`exceptions` - - field with those names will be skipped, recursively.

Returns:

this

Since:

5.8.0

---

isRequired

`bool` **`isRequired`**`()`

Is the field mandatory or not.

This is defined on the Validations tab of the field under the Designer.

Returns:

true if field is required field.

---

isRequiredOnSummaryOnly

`bool` **`isRequiredOnSummaryOnly`**`()`

Is the field mandatory only on summary or not.

This is defined on the Validations tab of the field under the Designer.

Returns:

true if field is required on summary only

---

calculate

`void` **`calculate`**`()`

Recalculates the field value.

This function has effect only on fields that have a calculated values, either through conditional values or through expressions entered on the Value tab for the field under the

Designer.

---

validate

`bool validate()`

Revalidates the field.

This method performs a revalidation of the rules as defined on the Validations tab of the field under the Designer.

Returns:
true if field is valid.

---

isValid

`bool isValid()`

Is the field valid or not.

Returns:
true if field is valid.

---

getErrorMessages

`string[] getErrorMessages()`

Gets the validation error messages of the field.

Returns:
error messages

---

getErrorCodes

`int[] getErrorCodes()`

Gets the validation error codes of the field. see also `Constants.ErrorCode`

Returns:

error codes

resetValidationResult

`void **resetValidationResult**()`

Cleans the validation results and error messages.

addTargetField

`void **addTargetField**()`

Add the current field as a target to itself, for autorefresh purposes.

Since:

7.0.0

addTargetField

`void **addTargetField**(string fieldId)`

Add a target field to the current field, for autorefresh purposes.

Parameters:

`fieldId` - - id of the field to be added

Since:

7.0.0

removeTargetField

`void **removeTargetField**()`

Removes the current field from itself as a target, for autorefresh purposes.

Since:

7.0.0

removeTargetField

```
void removeTargetField(string fieldId)
```

Removes the specified field from the current list of target fields for the current field, for autorefresh purposes.

Parameters:

`fieldId` - - id of the field to be removed

Since:

7.0.0

---

addTargetFieldByName

```
void addTargetFieldByName(string fieldName)
```

Add a target field to the current field, for autorefresh purposes.

Parameters:

`fieldName` - - name of the field to be added

Since:

7.0.0

---

removeTargetFieldByName

```
void removeTargetFieldByName(string fieldName)
```

Removes the specified field from the current list of target fields for the current field, for autorefresh purposes.

Parameters:

`fieldName` - - name of the field to be removed

Since:

7.0.0

---

addSourceField

```
void addSourceField(string fieldId)
```

Add a source field to the current field, for autorefresh purposes.

Parameters:

`fieldId` - - id of the field to be added

Since:

7.0.0

---

removeSourceField

```
void removeSourceField(string fieldId)
```

Removes the specified field from the current list of source fields for the current field, for autorefresh purposes.

Parameters:

`fieldId` - - id of the field to be removed

Since:

7.0.0

---

isAvailable

```
bool isAvailable()
```

Is the field available.

The concept of availability in SmartGuide refers to whether or not a field is relevant at a given point in the Smartlet. For example if a Smartlet has 4 pages, and a question on page 1 shows either page 2 or 3 depending on the outcome, then when arriving on page 4, the relevant (available) fields will be those which were on the pages visited to reach page 4. Assuming in that example that page 3 was not visited (because of an answer given on page 1), then all fields on page 3 will be considered as unavailable.

This concept extends also inside a given page. For example if a field inside a page has a visibility condition set, and that condition is not met, then it will be considered as non available.

It is also very important to note that a non available field cannot be mapped, for example on a PDF or XML file. Neither can it be mapped to the input of a service call.

Returns:

true if field is available

---

triggerEvent

```
void triggerEvent()
```

Triggers the event associated to a field. If the field is of type Button, the trigger will be a "click" event.

The following example demonstrates how to trigger the enter button for a subSmartlet.

```
ISmartlet smartlet = context.getSmartlet();
ISubSmartletField subsmartletfield = (ISubSmartletField) smartlet.findFieldByName(fieldName);
ISmartletField enterbutton = subsmartletfield.getEnterButton();
enterbutton.triggerEvent();
```

---

appendTo

```
void appendTo(ISmartletPage page,
              int position)
```

Appends field to page at given position.

Parameters:

`page` -

`position` -

Since:

5.4.0

---

appendTo

```
void appendTo(ISmartletField parent,
              int position)
```

Appends field under the parent field at the specified position . The parent field should be type of ISmartletGroup or ISmartletRepeat. A valid operation could be either appending ISmartletField to ISmartletGroup or appending ISmartletGroup to ISmartletRepeat. If current field is under repeat, all fields under repeat with same ID as current field will be moved. If parent field is under repeat, all the fields with the same ID will be moved to the corresponding position.

Parameters:

`parent` - - parent field

`position` - - at which the field is to be inserted. if the index is out of range (index < 0 || index > size()), the field will be added to the last.

Since:

5.4.0

---

appendBefore

```
void appendBefore(ISmartletField field)
```

Appends the field before given field. If current field is under repeat, all fields under repeat with same ID as current field will be moved. If parent field is under repeat, all the fields with the same ID will be moved to the corresponding position.

Parameters:

`field` -

Since:

5.4.0

---

appendAfter

```
void appendAfter(ISmartletField field)
```

Appends the field after given field. If current field is under repeat, all fields under repeat with same ID as current field will be moved. If parent field is under repeat, all the fields with the same ID will be moved to the corresponding position.

Parameters:

`field` -

Since:

5.4.0

---

detach

```
void detach()
```

Detaches a field. The detached field is placed on a special page of the smartlet. If current field is under repeat, all fields under repeat with same ID as current field will be detached.

Since:

5.4.0

---

applyDefinition

```
void applyDefinition()
```

Changes field definition.

Field definition includes:

◊ Field attributes: label, help, hint, css class, css width, css style, choice layout,max length, min length, placement, prefix, suffix, select options

◊ Field validations

◊ Field calculations

◊ Field availability

The change will take effect within users' session.

Since:

5.4.0

---

setValidationRequire

```
void setValidationRequire(string isRequiredScript,
                          string errorMessage)
```

Defines required validation.

Parameters:

`isRequiredScript` - - BSH script which returns true or false to determine if the field is required or not.

`errorMessage` - - error message when validations fails.

Since:

5.4.0

---

setValidationScript

```
void setValidationScript(string validationiScript)
```

Defines script validation

Parameters:

`validationiScript` - - BSH script to do the validation. If there is an error, the script returns the error message. If no error, script shall return null or empty string.

Since:

5.4.0

---

setValueParseScript

```
void setValueParseScript(string valueParseScript)
```

Defines script to parse value.

Parameters:

`valueParseScript` - - BSH script to to be called when getValue() method get called.

Since:

5.4.0

---

getValueParseScript

```
string getValueParseScript()
```

Gets script to parse the value

Returns:

- script used to parse the field value

---

resetValidationDefinitions

```
void resetValidationDefinitions()
```

Resets the validation definitions.

Since:

5.4.0

---

setValidateOnSummaryOnly

```
void setValidateOnSummaryOnly(bool bool)
```

If sets to true, the validation will happen only on summary section.

Parameters:

`bool` - - bool value

Since:

5.4.0

---

setAvailabilityScript

```
void setAvailabilityScript(string script)
```

Defines script to calculate field availability. If script returns bool value "true" or case-insensitive string "true" or 1, the field is available; otherwise the field is not available.

Parameters:

`script` - - BSH script to return true or false. If null or empty string, the availability definition is reset to default.

Since:

5.4.0

---

calculateAvailability

```
bool calculateAvailability()
```

Calculates and returns the field availability.

Returns:

true or false

Since:

5.4.0

---

setCalculationScript

```
void setCalculationScript(string script,
                          bool alwaysCalc)
```

Defines field calculation script.

Parameters:

`script` - - BSH script to do the field calculation

`alwaysCalc` - - if true, the value will always be calculated. If false, the value will be calculated only once.

Since:

5.4.0

---

getRepeatValues

```
Object[] getRepeatValues()
```

Returns values for repeated field. If field is not under repeat, returns array with one element.

Returns:

- value array

Since:

5.4.0

---

getRepeatSelectedValues

`Object[] getRepeatSelectedValues()`

Returns values for selected groups in repeated field. If field is not under repeat, returns array with one element.

Returns:

- value array

Since:

7.1.0

---

setRepeatValues

`void setRepeatValues(Object[] values)`

Sets the repeated field with values. The repeat instance number is adjusted according to the length of array.

Parameters:

`values` - - repeat field values

Since:

5.4.0

---

getRepeatStrings

`string[] getRepeatStrings()`

Returns strings for repeated field. If field is not under repeat, returns array with one element.

Returns:

- string array

Since:

5.4.0

---

setRepeatStrings

`void setRepeatStrings(string[] strings)`

Sets the repeated field with strings. The repeat instance number is adjusted according to the length of array.

Parameters:

`strings` - - repeat field strings

Since:

5.4.0

---

getRepeatSelectedStrings

```
string[] getRepeatSelectedStrings()
```

Returns strings for selected groups in repeated field. If field is not under repeat, returns array with one element.

Returns:

- string array

Since:

7.1.0

---

isShownInSummarySection

```
bool isShownInSummarySection()
```

Returns true if the field is shown under the summary section

Returns:

- bool

Since:

5.4.0

---

setError

```
void setError(string errorMessage)
```

Manually sets field error message. If errorMessage is null, it is the same as calling method resetValidationResult(). When the error message is not null, the field validation is ignored and field is always invalid until the error message is reset to null.

Parameters:

`errorMessage` -

Since:

5.4.0

---

getEventSource

`ISmartletField[] getEventSource()`

Returns an array of ISmartletField which have an impact, through validation rules, dynamic values or visibility conditions on the current field.

Parameters:

`errorMessage` -

Since:

5.4.1

---

getEventTarget

`ISmartletField[] getEventTarget()`

Returns an array of ISmartletField which are impacted, through validation rules, dynamic values or visibility conditions by the current field.

Parameters:

`errorMessage` -

Since:

5.4.1

---

clear

`void clear()`

Clears the fields. Groups will clear subfields and repeats will empty their rows and leave one empty instance.
If you want to fully empty a repeat, there is an overload on repeats that has an option for that (clear(Boolean))
Note: this method will not clean the uploaded files on the disk.

Since:

5.8.0

---

isUnderRepeatDefaultGroup

`bool isUnderRepeatDefaultGroup()`

Returns true if field is under repeat default group or is default group itself.

Returns:

bool

Since:

6.5.0

---

getLayoutAttribute

```
string getLayoutAttribute(string deviceName,
                                    string attributeName)
```

Get layout attribute string by device and attribute name. For bootstrap, the layout attribute names are: hidden, col, offset, push and pull The device names are xs, sm, md, lg For user defined layout, please refer to layout_info.json

Returns:

attributeName name of layout attribute

Since:

7.0.0

---

getLayoutAttributes

```
string getLayoutAttributes()
```

Get layout attributes string. For bootstrap, the layout attributes string is grid class string, Eg: "col-sm-12 col-md-6 hidden-xs col-sm-offset-1 col-md-pull-1 col-md-offset-1 col-md-push-1" For user defined layout, please refer to layout_info.json

Returns:

layout attribute string

Since:

7.0.0

---

setLayoutAttributes

```
void setLayoutAttributes(string layoutString)
```

Set layout attributes string. For bootstrap, the layout attributes string is grid class string, Eg: "col-sm-12 col-md-6 hidden-xs col-sm-offset-1 col-md-pull-1 col-md-offset-1 col-md-push-1" For user defined layout, please refer to layout_info.json

Note that if you apply this method on a field created dynamically, you must make sure to call the method after the field has been appended to the page or another control otherwise the attributes will be overwritten by the append method.

Parameters:

`layoutString` -

Since:

7.0.0

---

setPosition

```
void setPosition(int rowPos,
                 int colPos)
```

Set field position to new row, column of current layout (row and column starts with 0).

Each field takes one cell in the layout grid, the row and column number is the relative position of cell.
Eg: page contain 2 rows, each row has 2 fields, the fields' position value are

field(0,0) field(0,1)
field(1,0) field(1,1)

To position a field under group, need to append the field under the group first, then call this method to position the field within the group layout.
If rowPos is out of range, append new row to the end. If colPos is out of range, append new column to the end.

Parameters:

`rowPos` - : new row position starts with 0

`colPos` - : new column position starts with 0

Since:

6.6.0

---

getMetaDataNames

```
string[] getMetaDataNames()
```

Get all meta data names.

Returns:

all names of meta data of the field. Returns empty array if there is no meta data.

Since:

7.0.0

---

getMetaData

```
string getMetaData(string name)
```

Get meta data value by name.

Parameters:

`name` - IS case sensitive and cannot be null. 'foo', 'Foo' and 'FOO' are referring to different meta data names.

Returns:

value of the meta data name. Returns null if meta data is not defined.

Since:

7.0.0

---

setMetaData

```
void setMetaData(string name,
                 string value)
```

Set meta data.

Parameters:

`name` - IS case sensitive and cannot be null. 'foo', 'Foo' and 'FOO' are referring to different meta data names.

`value` - meta string value, null value will be replaced by empty string.

---

com.alphinat.sg5

Interface ISmartletEvent

```
public interface ISmartletEvent
```

`ISmartletEvent` is an interface representing a Smartlet event, like a click on a button.

Method Summary

`ISmartletEvent`	**`getParent`**`()` Gets the nested parent event
`ISmartletElement`	**`getSource`**`()` Gets the source `element` that fires this event.
`int`	**`getType`**`()` Gets the event type

Method Detail

getType

```
int getType()
```

Gets the event type

Returns:

type of event

See Also:

`Constants.SmartletEvent`

getSource

```
ISmartletElement getSource()
```

Gets the source `element` that fires this event.

Returns:

source element

---

getParent

```
ISmartletEvent getParent()
```

Gets the nested parent event

Returns:

parent event

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5

Interface ISmartletService

All Superinterfaces:

ISmartletElement

`public interface` **`ISmartletService`**
extends ISmartletElement

`ISmartletService` is an interface representing a Smartlet service and covers extension functions, SOAP and REST web services.

Method Summary

`bool`	**`call`**`()` Calls the service.
`bool`	**`call`**`(Object[] parameters)` Call the service with the provided parameters.
`void`	**`defineAttributeInputBehaviorOnNull`**`(string key, int level)` Defines the behavior for input attribute mapping on web services.
`void`	**`defineInputBehaviorOnNull`**`(string key, int level)` Defines the behavior for optional input mapping on web services.
`void`	**`defineInputDictionaryping`**`(string key, string script)` Defines service input mapping dynamically.
`void`	**`defineOutputDictionaryping`**`(string key, string script)` Defines service output mapping dynamically.
`string`	**`getError`**`()` Gets the error message if an error occured when calling the service.
`string`	**`getId`**`()` Obtains the unique internal identifier of the service.

`Object[]`	**`getDictionarypedParameters`**`()` Gets the mapped parameters according to the service input mappings.
`string`	**`getName`**`()` Gets the service name.
`Object`	**`getResult`**`()` Gets the service call result.
`Object`	**`getResult`**`(string key)` For web services, parameter "key" is xpath (namespace ignored).
`Object[]`	**`getResults`**`(string key)` For web services, parameter "key" is xpath (namespace ignored).

Methods inherited from interface com.alphinat.sg5.ISmartletElement

`accept, data, data, getDataNames, getTypeConst`

Method Detail

getName

`string` **`getName`**`()`

Gets the service name.

Returns:

service name

getId

`string` **`getId`**`()`

Obtains the unique internal identifier of the service.

Specified by:

`getId` in interface `ISmartletElement`

Returns:

service ID

getDictionarypedParameters

`Object[] getDictionarypedParameters()`

Gets the mapped parameters according to the service input mappings. For web services, it returns the XML string of the SOAP request. For extension function services, it returns and Array of parameters.

Returns:

input mapping results of service.

call

`bool call()`

Calls the service.

The following example shows how to iterate through the services, perform a call to a specific service, and get the result,

```
ISmartlet smartlet = context.getSmartlet();
ISmartletService[] services = smartlet.getServices();
ISmartletService service = null;
for(int i=0;i<services.Length;i++)
{
  service = services[i];
  if ("CurrencyLookupWS".Equals(service.getName()))
  {
    if (service.call())
    {
      String xmlResponse = (String)service.getResult();
    }
    break;
  }
}
```

Returns:

returns true if call is successful.

Throws:

`Exception`

call

```
bool call(Object[] parameters)
```

Call the service with the provided parameters. For web services, the parameter is a string containing a SOAP request body. For extension function services, the parameters are function parameters. if parameters is null, this function is equivalent to call().

Returns:

returns true if call is successful.

Throws:

`Exception`

---

getResult

```
Object getResult()
```

Gets the service call result. For web services, the return response is an XML string. For extension function services, the return is the function call result.

Returns:

result of service call.

---

getResult

```
Object getResult(string key)
```

For web services, parameter "key" is xpath (namespace ignored). Function returns single node value or a xpath evaluation result. For extension function services, the parameter "key" is ignored. Function returns the service call result.

Parameters:

`key` - - xpath of web service.

Returns:

service result

---

getResults

```
Object[] getResults(string key)
```

For web services, parameter "key" is xpath (namespace ignored). Function returns node set values or a xpath evaluation results set. For extension function services, the parameter "key" is ignored. Function returns the object array which only has one element - the service call result.

Parameters:

`key` - - xpath of web service.

Returns:

results array

---

getError

```
string getError()
```

Gets the error message if an error occured when calling the service.

Returns:

error message if error occurs

---

defineInputDictionaryping

```
void defineInputDictionaryping(string key,
                               string script)
```

Defines service input mapping dynamically. For extension function service, the key will be index "1", "2", ... etc, starting from 1. For web service, the key is the xpath pointing to the response XML without name space. If script is null, the mapping is removed.

Parameters:

`key` - - key of mapping

`script` - - BSH script

Since:

5.4.0

---

defineOutputDictionaryping

```
void defineOutputDictionaryping(string key,
                                string script)
```

Defines service output mapping dynamically. If script is null, the mapping is removed.

Parameters:

`key` - - key of mapping, it is either field name or field id.

`script` - - BSH script

Since:

5.4.0

---

defineInputBehaviorOnNull

```
void defineInputBehaviorOnNull(string key,
                               int level)
```

Defines the behavior for optional input mapping on web services. The key is the xpath pointing to the response XML without name space. The level specifies at which point the optional node will be sent.

Parameters:

`key` - - Key of mapping

`level` - - Level, See `Constants.WSInputBehavior`

Since:

5.5.1

---

defineAttributeInputBehaviorOnNull

```
void defineAttributeInputBehaviorOnNull(string key,
                                        int level)
```

Defines the behavior for input attribute mapping on web services. The key is the xpath pointing to the response XML without name space. The level specifies at which point the null or empty attribute will be sent.

Parameters:

`key` - - Key of mapping

`level` - - Level, See `Constants.WSInputBehavior`

Since:

5.6.0

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**
PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5

Interface ISmartletElementVisitor

```
public interface ISmartletElementVisitor
```

Implements the Hierarchical Visitor Pattern to traverse Smartlet elements. The order of visit of elements within a Smartlet is:

```
Smartlet.
  Page 1
      Field 1
      Group 2
            Field 2-1
            ...
      Repeat 3
            Field 3-1
            ...
            Repeat 3 add button.
            Repeat 3 delete button.
     Page 1 navigation buttons
  ....
  Page 2
  ....
  Page N
  Global navigation buttons
  Subsmart related buttons
  Smartlet services
```

Please note that elements under subsmartlet are not visited.

Compared to the traditional VisitorPattern, the Hierarchical Visitor Pattern improves by:

1. hierarchical navigation -- the traditional VisitorPattern has no concept of depth. As a result, visitor cannot determine if one composite is within another composite or beside
2. conditional navigation -- the traditional VisitorPattern does not allow branches to be skipped. As a result the visitor cannot stop, filter, or optimize traversal based on some condition.

Hierarchical navigation

Hierarchical navigation is important for any traversal that needs to know whether one node is the child of another or its sibling. The simplest example of this is tree listings where an indentation level needs to be maintained. With the traditional VisitorPattern, one can only determine when we are entering a node. This tells us that we may want to indent but gives us no clues about outdenting. We don't know if we have left the previous node before we entered the current node.

The HierarchicalVisitorPattern removes this limitation by defining a two method protocol when visiting nodes -- **visitEnter** and **visitLeave**. If we are entering the node *bar* before leaving the node *foo*, we can safely assume that *bar* is a child (and not a sibling) of the composite *foo*.

Conditional Navigation

Conditional navigation is required to skip unnecessary branches and all of their children. Consider the second operation of the File System example. The search for a specific file in a particular path can only be performed optimally if we can dispose of branches that have no possibility of providing a match. Consider the following graph:

```
*** Page1.
****** Group1.1
****** Group1.2
********* Field 1.2.1
****** Repeat1.3
********* Field 1.3.1
********* Field 1.3.2

*** Page 2.
****** Group 2.1
****** Group 2.2
```

The traditional VisitorPattern would have to visit each leaf of the entire structure in order to find the Group 2.2. Even though we can see that Page1 does not match the root ancestor of Group2.2;, we would still have no choice but to perform a match for the leaf "1.3.1.1". The only way to avoid this is to abandon the traditional visitor and use another means of traversal. Most programmers violate the encapsulation provided by the traditional visitor when performing tree searches.

HierarchicalVisitorPattern allows us to solve this problem within a single visiting paradigm. It does so by having each invocation of *accept* answer a bool traversal state for its depth of the tree. For example, if *accept* on a composite or leaf answers *false*, traversal immediately stops at that tree depth. In other words, no more of its siblings will be traversed, even if some of those siblings are composites with children of their own. Reconsider the example graph. As we visit the node labeled "1", we can cause its *accept* message to answer *false* like so:

```
bool visitEnter( ISmartletElement ele ){
      if (ele is ISmartletPage){
              ISmartletPage page = (ISmartletPage)ele;
```

```
                return page.getName().Equals("Page 2");
        } else {
                return true;
        }
    }
```

If the page is not "Page 2", we do not enter the node and we do not traverse its children. We then proceed directly to "Page 2".

Method Summary

`bool`	**`visit`**`(ISmartletElement smartletElement)` Visit the Smartlet element.
`bool`	**`visitEnter`**`(ISmartletElement smartletElement)` Notifies the visitor that it is entering a new element.
`bool`	**`visitLeave`**`(ISmartletElement smartletElement)` Notifies the visitor that the element is visited.

Method Detail

visitEnter

`bool visitEnter(ISmartletElement smartletElement)`

Notifies the visitor that it is entering a new element.

Parameters:

`smartletElement` -

Returns:

if visitEnter return true, accept is invoked on each of its children or until one of the accept invocations answers false.

visitLeave

```
bool visitLeave(ISmartletElement smartletElement)
```

Notifies the visitor that the element is visited. if visitLeave returns false, this would short-circuit visiting its sibling nodes.

Parameters:

`smartletElement` -

Returns:

true if continue to visit.

---

visit

```
bool visit(ISmartletElement smartletElement)
```

Visit the Smartlet element.

Parameters:

`smartletElement` -

Returns:

true if continue to visit.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5

Interface ISmartletElement

All Known Subinterfaces:

ISmartlet, ISmartletDate, ISmartletField, ISmartletGroup, ISmartletKnowledge, ISmartletPage, ISmartletRepeat, ISmartletSelectField, ISmartletService, ISmartletSummary, ISmartletUpload, ISubSmartletField

`public interface **ISmartletElement**`

A Smartlet element can be a `ISmartlet`, `ISmartletPage`, `ISmartletField`, `ISmartletService`.

It is often used in the context of the `visitor pattern` to filter out elements in a Smartlet for processing. Refer to the examples provided in the Developer's guide.

Method Summary

`bool`	**`accept`**`(ISmartletElementVisitor visitor)` Implements the visitor's pattern to traverse Smartlet/page/field/services
`Object`	**`data`**`(string key)` Returns stored data of smartlet element.
`void`	**`data`**`(string key, Object value)` Stores data to smartlet element.
`string[]`	**`getDataNames`**`()` Returns names of data stored.
`string`	**`getId`**`()` Obtains the unique internal identifier of a Smartlet element.
`int`	**`getTypeConst`**`()` Gets the type of Smartlet element.

Method Detail

getId

`string getId()`

Obtains the unique internal identifier of a Smartlet element.

Returns:

id of Smartlet element.

---

getTypeConst

`int getTypeConst()`

Gets the type of Smartlet element.

Returns:

type of Smartlet element.

---

accept

`bool accept(ISmartletElementVisitor visitor)`

Implements the visitor's pattern to traverse Smartlet/page/field/services

Parameters:

`visitor` -

Returns:

true if continue to visit.

---

data

`Object data(string key)`

Returns stored data of smartlet element.

Parameters:

`key` - - Name of the data stored

Returns:

stored data

Since:

5.4.0

---

data

```
void data(string key,
          Object value)
```

Stores data to smartlet element.

Parameters:

`key` - - Name of the data stored

`value` - - stored data, data removed if it is null.

Since:

5.4.0

---

getDataNames

```
string[] getDataNames()
```

Returns names of data stored.

Returns:

- array of name string

Since:

5.4.0

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5

Interface ISmartletActionError

```
public interface ISmartletActionError
```

Method Summary

`string`	`getError()` Obtains the error message.
`string`	`getSource()` Obtains the error source.
`string`	`getStackTrace()` Obtains the stack trace.

Method Detail

getError

```
string getError()
```

Obtains the error message.

Returns:

the `string` representing the error message as returned by the action

getSource

```
string getSource()
```

Obtains the error source.

Returns:

the `string` representing the class which, when executed encountered an error

---

getStackTrace

```
string getStackTrace()
```

Obtains the stack trace.

Returns:

the `string` representing the stack trace where the error was logged.

---

com.alphinat.sg5.widget.summary

Interface ISmartletSummary

All Superinterfaces:

ISmartletElement, ISmartletField

`public interface` **`ISmartletSummary`**
extends ISmartletField

`ISmartletSummary` is an interface representing a summary section.

Method Summary

`ISmartletPage[]`	**`getShownPages`**`()` Gets shown pages undr the summary section.
`bool`	**`isElementExcluded`**`(ISmartletElement smartletElement)` Returns true if a page or field is excluded from the summary section.

Methods inherited from interface com.alphinat.sg5.ISmartletField

`addSourceField, addTargetField, addTargetField, addTargetFieldByName, appendAfter, appendBefore, appendTo, appendTo, applyDefinition, calculate, calculateAvailability, clear, detach, getChoiceLayout, getCSSClass, getCSSHeight, getCSSStyle, getCSSWidth, getErrorCodes, getErrorMessages, getEventSource, getEventTarget, getFormat, getHelp, getHelpId, getHtmlName, getId, getLabel, getLayoutAttribute, getLayoutAttributes, getMaxLength, getMetaData, getMetaDataNames, getMinLength, getName, getNext, getPage, getParent, getPlacement, getPrefix, getPrevious, getRepeatIndex, getRepeatSelectedStrings, getRepeatSelectedValues, getRepeatStrings, getRepeatValues, getSmartlet, getString, getSuffix, getTooltip, getTypeConst, getTypeDetail, getValue, getValueParseScript, isAvailable, isEncrypted, isHelpLink, isPersistent, isReadonly, isRequired, isRequiredOnSummaryOnly, isShownInSummarySection,` isUnderRepeatDefaultGroup, isValid, removeSourceField, removeTargetField, removeTargetField, removeTargetFieldByName, resetValidationDefinitions, resetValidationResult, setAvailabilityScript, setCalculationScript, setChoiceLayout, setCSSClass, setCSSHeight, setCSSStyle, setCSSWidth, setEncrypted, setError, setFormat, setHelp, setLabel, setLayoutAttributes, setMetaData, setPersistent, setPlacement, setPosition, setPrefix, setReadonly, setReadonly, setRepeatStrings, setRepeatValues, setString, setSuffix, setTooltip, setValidateOnSummaryOnly, setValidationRequire, setValidationScript, setValue, setValueParseScript, triggerEvent, validate

Methods inherited from interface com.alphinat.sg5.ISmartletElement

accept, data, data, getDataNames

Method Detail

isElementExcluded

```
bool isElementExcluded(ISmartletElement smartletElement)
```

Returns true if a page or field is excluded from the summary section.

Parameters:

`smartletElement` - - can be a ISmartletPage of ISmartletField

Returns:

- bool value

getShownPages

```
ISmartletPage[] getShownPages()
```

Gets shown pages undr the summary section.

Returns:

array of `ISmartletField` on the page

Since:

5.4.0

com.alphinat.sg5.widget.subsmartlet

Interface ISubSmartletField

All Superinterfaces:

ISmartletElement, ISmartletField

`public interface` **`ISubSmartletField`**
extends ISmartletField

`ISubSmartletField` is an interface representing a subSmartlet.

Method Summary

`ISmartletField`	**`getEnterButton`**`()` Gets a button to enter the subSmartlet.
`string`	**`getSubSmartletCode`**`()` Gets the subSmartlet code.
`void`	**`init`**`()` Initializes a subSmartlet.

Methods inherited from interface com.alphinat.sg5.ISmartletField

`addSourceField, addTargetField, addTargetField, addTargetFieldByName, appendAfter, appendBefore, appendTo, appendTo, applyDefinition, calculate, calculateAvailability, clear, detach, getChoiceLayout, getCSSClass, getCSSHeight, getCSSStyle, getCSSWidth, getErrorCodes, getErrorMessages, getEventSource, getEventTarget, getFormat, getHelp, getHelpId, getHtmlName, getId, getLabel, getLayoutAttribute, getLayoutAttributes, getMaxLength, getMetaData, getMetaDataNames, getMinLength, getName, getNext, getPage, getParent, getPlacement, getPrefix, getPrevious, getRepeatIndex, getRepeatSelectedStrings, getRepeatSelectedValues, getRepeatStrings, getRepeatValues, getSmartlet, getString, getSuffix, getTooltip, getTypeConst, getTypeDetail, getValue, getValueParseScript, isAvailable,` isEncrypted, isHelpLink, isPersistent, isReadonly, isRequired, isRequiredOnSummaryOnly, isShownInSummarySection, isUnderRepeatDefaultGroup, isValid, removeSourceField, removeTargetField, removeTargetField, removeTargetFieldByName, resetValidationDefinitions, resetValidationResult, setAvailabilityScript, setCalculationScript, setChoiceLayout, setCSSClass, setCSSHeight, setCSSStyle, setCSSWidth, setEncrypted, setError, setFormat, setHelp, setLabel, setLayoutAttributes, setMetaData, setPersistent, setPlacement, setPosition, setPrefix, setReadonly, setReadonly, setRepeatStrings, setRepeatValues, setString, setSuffix, setTooltip, setValidateOnSummaryOnly, setValidationRequire, setValidationScript, setValue, setValueParseScript, triggerEvent, validate

Methods inherited from interface com.alphinat.sg5.ISmartletElement

accept, data, data, getDataNames

Method Detail

init

```
void init()
```

Initializes a subSmartlet.

Before accessing a subSmartlet field's fields, it must be initialized, unless it has already been visited once. The following example shows how to do that. It assumes that a subSmartlet field named "personalInfo" is available.

```
ISubSmartletField subsmartletfield = (ISubSmartletField)smartlet.findFieldByName("personalInfo");
ISmartlet subsmartlet = (ISmartlet)subsmartletfield.getValue();
if (subsmartlet == null)
{
   subsmartletfield.init();
   subsmartlet = (ISmartlet)subsmartletfield.getValue();
}
```

As observed in the example, it is necessary to get the value of the subSmartlet field to access the subSmartlet instance. Because the field itself belongs to the main Smartlet.

getSubSmartletCode

```
string getSubSmartletCode()
```

Gets the subSmartlet code.

Returns:

subSmartlet code

getEnterButton

```
ISmartletField getEnterButton()
```

Gets a button to enter the subSmartlet.

The following example demonstrates how to trigger the enter button for a subSmartlet.

```
ISmartlet smartlet = context.getSmartlet();
ISubSmartletField subsmartletfield = (ISubSmartletField) smartlet.findFieldByName(fieldName);
ISmartletField enterbutton = subsmartletfield.getEnterButton();
enterbutton.triggerEvent();
```

Returns:

button

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5.widget.select

Interface ISmartletSelectField

All Superinterfaces:

ISmartletElement, ISmartletField

`public interface` **`ISmartletSelectField`**
extends ISmartletField

`ISmartletField` is an interface representing a Smartlet select type field. These can be radio button, dropdown list, checkbox or listbox.

Method Summary

`string`	**`getSelectedLabel`**`()` Returns label of selected option.
`string[]`	**`getSelectedLabels`**`()` Returns labels of selected options.
`ISelectOption`	**`getSelectedOption`**`()` Returns selected option.
`ISelectOption[]`	**`getSelectedOptions`**`()` Returns array of selected options.
`ISelectOptionList`	**`getSelectOptions`**`()` Returns the option list for the select type field.
`void`	**`setOptions`**`(Object[] labels, Object[] values)` Sets select options with labels and values.

Methods inherited from interface com.alphinat.sg5.ISmartletField

`addSourceField, addTargetField, addTargetField, addTargetFieldByName, appendAfter, appendBefore, appendTo, appendTo, applyDefinition, calculate, calculateAvailability,` clear, detach, getChoiceLayout, getCSSClass, getCSSHeight, getCSSStyle, getCSSWidth, getErrorCodes, getErrorMessages, getEventSource, getEventTarget, getFormat, getHelp, getHelpId, getHtmlName, getId, getLabel, getLayoutAttribute, getLayoutAttributes, getMaxLength, getMetaData, getMetaDataNames, getMinLength, getName, getNext, getPage, getParent, getPlacement, getPrefix, getPrevious, getRepeatIndex, getRepeatSelectedStrings, getRepeatSelectedValues, getRepeatStrings, getRepeatValues, getSmartlet, getString, getSuffix, getTooltip, getTypeConst, getTypeDetail, getValue, getValueParseScript, isAvailable, isEncrypted, isHelpLink, isPersistent, isReadonly, isRequired, isRequiredOnSummaryOnly, isShownInSummarySection, isUnderRepeatDefaultGroup, isValid, removeSourceField, removeTargetField, removeTargetField, removeTargetFieldByName, resetValidationDefinitions, resetValidationResult, setAvailabilityScript, setCalculationScript, setChoiceLayout, setCSSClass, setCSSHeight, setCSSStyle, setCSSWidth, setEncrypted, setError, setFormat, setHelp, setLabel, setLayoutAttributes, setMetaData, setPersistent, setPlacement, setPosition, setPrefix, setReadonly, setReadonly, setRepeatStrings, setRepeatValues, setString, setSuffix, setTooltip, setValidateOnSummaryOnly, setValidationRequire, setValidationScript, setValue, setValueParseScript, triggerEvent, validate

Methods inherited from interface com.alphinat.sg5.ISmartletElement

accept, data, data, getDataNames

Method Detail

getSelectOptions

`ISelectOptionList getSelectOptions()`

Returns the option list for the select type field.

Returns:

a new ISelectOptionList

getSelectedOptions

`ISelectOption[] getSelectedOptions()`

Returns array of selected options.

Returns:

- ISelectOption[]

Since:

5.4.0

---

getSelectedOption

`ISelectOption getSelectedOption()`

Returns selected option.

Returns:

- ISelectOption, null if no selected option.

Since:

5.4.0

---

getSelectedLabels

`string[] getSelectedLabels()`

Returns labels of selected options.

Returns:

- string array, returns empty array if no selected option

Since:

5.4.0

---

getSelectedLabel

`string getSelectedLabel()`

Returns label of selected option.

Returns:

- string array, null if no selected option.

Since:

5.4.0

---

setOptions

```
void setOptions(Object[] labels,
                Object[] values)
```

Sets select options with labels and values.

Parameters:

`labels` -

`values` -

Since:

5.4.0

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5.widget.select

Interface ISelectOptionList

`public interface ISelectOptionList`

`ISelectOptionList` is an interface representing the single or multiple select option list associated to a select type field. Each list item is an instance of `ISelectOption`. Manipulating the list directly affects the corresponding field's option item list. A field's option list may be obtained by using the `ISmartletSelectField.getSelectOptions()` functionality.

Method Summary

`void`	**`add`**`(int index, Object element)` Inserts the specified element at the specified position in this list.
`bool`	**`add`**`(Object o)` Appends the specified element to the end of this list.
`void`	**`clear`**`()` Removes all of the elements from this selection item list.
`ISelectOption`	**`createOption`**`()` Creates a new `ISelectOption` instance that can be added to the list.
`ISelectOption`	**`createOptionGroup`**`()` Creates an option group.
`Object`	**`get`**`(int index)` Returns the item at the specified position in this list.
`bool`	**`isEmpty`**`()` Returns `true` if this selection item list contains no elements.
`Object`	**`remove`**`(int index)` Removes the element at the specified position in this list.
`Object`	**`set`**`(int index, Object element)` Replaces the item at the specified position in this list with the specified

	element.
`int`	`size()` Returns the number of items in this selection item list.

Method Detail

createOption

`ISelectOption createOption()`

Creates a new `ISelectOption` instance that can be added to the list. It is to be noted that only these instances are allowed to be inserted into the list.

Returns:

a new `ISelectOption` instance that can be added to the list.

createOptionGroup

`ISelectOption createOptionGroup()`

Creates an option group.

Returns:

a new `ISelectOption` instance which isOptionGroup returns true.

Since:

6.6.0

size

`int size()`

Returns the number of items in this selection item list.

Returns:

the number of items in this selection item list.

isEmpty

`bool **isEmpty**()`

Returns `true` if this selection item list contains no elements.

Returns:

`true` if this selection item list contains no elements.

---

add

`bool **add**(Object o)`

Appends the specified element to the end of this list.

Parameters:

`o` - element to be appended to this selection item list.It is to be noted that only `ISelectOption` are allowed to be added.

Returns:

`true` if the item is successfully added.

---

clear

`void **clear**()`

Removes all of the elements from this selection item list. The list will be empty after this call returns.

---

get

`Object **get**(int index)`

Returns the item at the specified position in this list.

Parameters:

`index` - index, 0-based, of item to return.

Returns:

the item at the specified position in this list.

---

set

```
Object set(int index,
                  Object element)
```

Replaces the item at the specified position in this list with the specified element.

Parameters:

`index` - index, 0-based, of item to replace.

`element` - item to be stored at the specified position.

Returns:

the item previously at the specified position.

---

add

```
void add(int index,
         Object element)
```

Inserts the specified element at the specified position in this list. Shifts the element currently at that position (if any) and any subsequent elements to the right (adds one to their indices).

Parameters:

`index` - index, 0-based, at which the specified element is to be inserted.

`element` - element to be inserted.

---

remove

```
Object remove(int index)
```

Removes the element at the specified position in this list. Shifts any subsequent elements to the left (subtracts one from their indices). Returns the element that was removed from the list.

Parameters:

`index` - the index, 0-based, of the element to removed.

Returns:

the element previously at the specified position.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5.widget.select

Interface ISelectOption

`public interface` **`ISelectOption`**

`ISelectOption` is an interface representing a single or multiple selection list item. Individual item attributes may be manipulated. New list item instances can be created using the `ISelectOptionList.createOption()` functionality.

Method Summary

`string`	**`getHelp`**`()` Gets the help text of this selection item.
`string`	**`getHelpId`**`()` Gets the help id used to render help link
`string`	**`getHint`**`()` Gets the hint text of this selection item.
`string`	**`getLabel`**`()` Gets the label of this selection item.
`ISelectOptionList`	**`getSubOptions`**`()` Gets the sub options if this is an option group.
`string`	**`getValue`**`()` Gets the value of this selection item.
`bool`	**`isLink`**`()` Returns true if the help is a link.
`bool`	**`isOptionGroup`**`()` Returns true if the option is an option group.
`void`	**`setHelp`**`(string help)` Sets the help for this selection item to the specified help.

`void`	**`setHint`**`(string hint)` Sets the hint for this selection item to the specified hint.
`void`	**`setLabel`**`(string label)` Sets the label for this selection item to the specified label.
`void`	**`setLink`**`(bool isLink)` Sets whether the help text is a link.
`void`	**`setOptionGroup`**`(bool isOptionGroup)` Sets true if it is option group.
`void`	**`setValue`**`(string value)` Sets the value for this selection item to the specified value.

Method Detail

getLabel

```
string getLabel()
```

Gets the label of this selection item.

Returns:

the label of this selection item

setLabel

```
void setLabel(string label)
```

Sets the label for this selection item to the specified label.

Parameters:

`label` - - option label.

getValue

```
string getValue()
```

Gets the value of this selection item.

Returns:
the value of this selection item

---

setValue

`void **setValue**(string value)`

Sets the value for this selection item to the specified value.

Parameters:
`value` - - option value string.

---

getHelp

`string **getHelp**()`

Gets the help text of this selection item.

Returns:
the help of this selection item

---

getHelpId

`string **getHelpId**()`

Gets the help id used to render help link

Returns:
help id to render link to help.

Since:
5.4.0

---

setHelp

`void **setHelp**(string help)`

Sets the help for this selection item to the specified help.

Parameters:

`help` - - help text.

---

getHint

```
string getHint()
```

Gets the hint text of this selection item.

Returns:

the hint of this selection item

---

setHint

```
void setHint(string hint)
```

Sets the hint for this selection item to the specified hint.

---

isLink

```
bool isLink()
```

Returns true if the help is a link.

Returns:

true if the help is a link

---

setLink

```
void setLink(bool isLink)
```

Sets whether the help text is a link.

Parameters:

`isLink` - - if true, the help text is a link.

---

setOptionGroup

```
void setOptionGroup(bool isOptionGroup)
```

Sets true if it is option group.

Since:

6.6.0

---

isOptionGroup

```
bool isOptionGroup()
```

Returns true if the option is an option group.

Returns:

if the option is an option group

---

getSubOptions

```
ISelectOptionList getSubOptions()
```

Gets the sub options if this is an option group. Return null if this is not an option group.

Returns:

`ISelectOptionList` for option group.

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5.widget.repeat

Interface ISmartletRepeat

All Superinterfaces:

ISmartletElement, ISmartletField

`public interface` **`ISmartletRepeat`**
extends ISmartletField

`ISmartletRepeat` is an interface representing a Smartlet repeat widget. A repeat widget can be thought of as a table. Each row of the table is a group. And each column of the table is a field in the group.

Method Summary

`ISmartletGroup`	**`addGroup`**`()` Creates an empty group and add it to the end.
`ISmartletGroup`	**`addGroup`**`(int position)` Create an empty group and insert it to the given position.
`ISmartletGroup`	**`addGroup`**`(int position, ISmartletGroup group)` Adds a repeat group to the specified position.
`ISmartletGroup`	**`addGroup`**`(ISmartletGroup group)` Adds a repeat group to the end.
`void`	**`clear`**`()` Clears the repeat instances, leaves one empty instance.
`ISmartletRepeat`	**`clear`**`(java.lang.Boolean leaveEmptyInstance)` **Deprecated.** *since 6.5.0*
`ISmartletGroup`	**`createDetachedGroup`**`()` Creates a repeatable group instance.
`ISmartletRepeat`	**`filterByColumn`**`(string columnName, string subString, bool ignoreCase)`

	Filter the repeat's groups, keeping only those containing given substring in given column.
`ISmartletField[]`	**`findAllFields`**`()` Returns all fields under the repeat.
`ISmartletField[]`	**`findFieldsById`**`(string id)` Finds all matching fields by id inside the repeat.
`ISmartletField[]`	**`findFieldsByName`**`(string name)` Finds all fields by name inside the repeat.
`ISmartletField[]`	**`findFieldsByRegex`**`(string regularExpression)` Finds fields by regulation expression under the repeat.
`ISmartletField[]`	**`findFieldsByScript`**`(string script)` Finds fields by matching script under the repeat.
`ISmartletField[]`	**`findFieldsByTypes`**`(int[] types)` Finds fields by types under the repeat.
`int`	**`getCount`**`()` Gets a count of repeated instances.
`ISmartletGroup`	**`getDefaultGroup`**`()` Returns default group as template.
`ISmartletGroup`	**`getGroup`**`(int position)` Returns the group at the specified position in the repeat.
`ISmartletGroup[]`	**`getGroups`**`()` Returns the groups of fields for the repeat.
`int[]`	**`getSelectedGroupIndexes`**`()` Gets an integer array of the selected rows.
`ISmartletGroup[]`	**`getSelectedGroups`**`()` Returns the selected groups of fields for the repeat.
`int[]`	**`getUnSelectedGroupIndexes`**`()` Gets an integer array of the non selected rows.
`ISmartletGroup[]`	**`getUnSelectedGroups`**`()` Returns the non selected groups of fields for the repeat.
`ISmartletRepeat`	**`moveDown`**`(ISmartletGroup group)` Move given group down a row.

ISmartletRepeat	**moveFirst**(ISmartletGroup group) Move given group to the top of the repeat.
ISmartletRepeat	**moveLast**(ISmartletGroup group) Move given group to the bottom of the repeat.
ISmartletRepeat	**moveUp**(ISmartletGroup group) Move given group up a row.
ISmartletGroup	**removeGroup**(int position) Removes the group at the specified position in the repeat.
ISmartletGroup	**setGroup**(int position, ISmartletGroup group) Sets a group to a specific position in the repeat.
ISmartletRepeat	**setReadonly**(bool readonly) Sets the "readonly" flag for all fields in all groups of this repeat.
ISmartletRepeat	**sortByColumn**(string columnName) Sorts groups of repeat alphanumerically with given field's string value.
ISmartletRepeat	**sortByColumnDescending**(string columnName) Sorts groups of repeat alphanumerically with given field's string value.

Methods inherited from interface com.alphinat.sg5.ISmartletField

addSourceField, addTargetField, addTargetField, addTargetFieldByName, appendAfter, appendBefore, appendTo, appendTo, applyDefinition, calculate, calculateAvailability, detach, getChoiceLayout, getCSSClass, getCSSHeight, getCSSStyle, getCSSWidth, getErrorCodes, getErrorMessages, getEventSource, getEventTarget, getFormat, getHelp, getHelpId, getHtmlName, getId, getLabel, getLayoutAttribute, getLayoutAttributes, getMaxLength, getMetaData, getMetaDataNames, getMinLength, getName, getNext, getPage, getParent, getPlacement, getPrefix, getPrevious, getRepeatIndex, getRepeatSelectedStrings, getRepeatSelectedValues, getRepeatStrings, getRepeatValues, getSmartlet, getString, getSuffix, getTooltip, getTypeConst, getTypeDetail, getValue, getValueParseScript, isAvailable, isEncrypted, isHelpLink, isPersistent, isReadonly, isRequired, isRequiredOnSummaryOnly, isShownInSummarySection, isUnderRepeatDefaultGroup, isValid, removeSourceField, removeTargetField, removeTargetField, removeTargetFieldByName, resetValidationDefinitions, resetValidationResult, setAvailabilityScript, setCalculationScript, setChoiceLayout, setCSSClass, setCSSHeight, setCSSStyle, setCSSWidth, setEncrypted, setError, setFormat, setHelp, setLabel, setLayoutAttributes, setMetaData, setPersistent, setPlacement, setPosition, setPrefix, setReadonly, setRepeatStrings, setRepeatValues, setString, setSuffix, setTooltip, setValidateOnSummaryOnly, setValidationRequire, setValidationScript, setValue, setValueParseScript, triggerEvent, validate

Methods inherited from interface com.alphinat.sg5.ISmartletElement

accept, data, data, getDataNames

Method Detail

createDetachedGroup

`ISmartletGroup` **`createDetachedGroup`**`()`

Creates a repeatable group instance. This group is detached from the repeat.

This method can be used to quickly get an instance of the repeatable group, modify its content, and then add it to the repeatable instance via addGroup(ISmartletGroup).

Returns:

newly created group which is detached.

---

addGroup

`ISmartletGroup` **`addGroup`**`()`

Creates an empty group and add it to the end.

Returns:

new empty group added to the end.

---

addGroup

```
ISmartletGroup addGroup(int position)
```

Create an empty group and insert it to the given position.

Parameters:

`position` - - position, 0-based, to insert the new group.

Returns:

newly created group inserted into the given position.

---

addGroup

```
ISmartletGroup addGroup(ISmartletGroup group)
```

Adds a repeat group to the end. Only has effect for repeat fields.

Parameters:

`group` - - the group to add.

Returns:

newly created

---

addGroup

```
ISmartletGroup addGroup(int position,
                        ISmartletGroup group)
```

Adds a repeat group to the specified position.

Parameters:

`position` - - position to insert

`group` - - group to be added.

Returns:

the added group.

---

setGroup

```
ISmartletGroup setGroup(int position,
                        ISmartletGroup group)
```

Sets a group to a specific position in the repeat.

Parameters:

`position` - - position, 0-based, to replace

`group` - - group to be placed

Returns:

group to replace.

---

removeGroup

`ISmartletGroup removeGroup(int position)`

Removes the group at the specified position in the repeat.

Parameters:

`position` - - position, 0-based, to remove.

Returns:

removed group.

---

getGroups

`ISmartletGroup[] getGroups()`

Returns the groups of fields for the repeat.

Returns:

repeat groups.

---

getSelectedGroups

`ISmartletGroup[] getSelectedGroups()`

Returns the selected groups of fields for the repeat.

Returns:

repeat groups.

Since:

7.1.0

---

getUnSelectedGroups

`ISmartletGroup[] getUnSelectedGroups()`

Returns the non selected groups of fields for the repeat.

Returns:
repeat groups.
Since:
7.1.0

---

getGroup

`ISmartletGroup getGroup(int position)`

Returns the group at the specified position in the repeat. Position can be negative number, it means that you count from the right instead of the left. getGroup(-1) refers to the last element, getGroup(-2) is the second-last, and so on.

Parameters:
`position` - - position, 0-based, can be negative.
Returns:
repeat group at position, returns null if position is invalid or out of range.
Since:
6.5.0

---

getDefaultGroup

`ISmartletGroup getDefaultGroup()`

Returns default group as template. This group is read only.
Returns:
default group
Since:
6.5.0

---

getCount

`int getCount()`

Gets a count of repeated instances.

Returns:

repeat group count.

---

clear

```
void clear()
```

Clears the repeat instances, leaves one empty instance. Note: this method will not clean the uploaded files on the disk. Unlike clicking the delete button, no validation will be performed.

Specified by:

`clear` in interface `ISmartletField`

Since:

5.8.0

---

clear

```
ISmartletRepeat clear(java.lang.Boolean leaveEmptyInstance)
```

Deprecated. *since 6.5.0*

Clears the repeat instances. Will leave one empty instance if specified. Note: this method will not clean the uploaded files on the disk. Unlike clicking the delete button, no validation will be performed. This method is @deprecated, Repeat has default group as template instance.

Parameters:

`leaveEmptyInstance` -

Since:

5.8.0

---

findFieldsByName

```
ISmartletField[] findFieldsByName(string name)
```

Finds all fields by name inside the repeat.

Parameters:

`name` - - Field name.

Returns:

`array of Smartlet fields`

---

findFieldsById

`ISmartletField[] findFieldsById(string id)`

Finds all matching fields by id inside the repeat.

Parameters:

`id` -

Returns:

`array of Smartlet fields`

---

findFieldsByTypes

`ISmartletField[] findFieldsByTypes(int[] types)`

Finds fields by types under the repeat.

Parameters:

`types` - - field types to find

Returns:

`smartlet fields`

Since:

5.4.0

---

findFieldsByRegex

`ISmartletField[] findFieldsByRegex(string regularExpression)`

Finds fields by regulation expression under the repeat.

Parameters:

`regularExpression` - - regulation expression to match the field name

Returns:

`array of Smartlet fields`

Since:

5.4.0

---

findFieldsByScript

`ISmartletField[] findFieldsByScript(string script)`

Finds fields by matching script under the repeat.

Parameters:

`script` - - BSH script

Returns:

`array of Smartlet fields`

Since:

5.4.0

---

findAllFields

`ISmartletField[] findAllFields()`

Returns all fields under the repeat.

Returns:

`array of Smartlet fields`

---

moveUp

`ISmartletRepeat moveUp(ISmartletGroup group)`

Move given group up a row.

Parameters:

`group` -

Returns:

this

Since:

5.8.0

---

moveDown

`ISmartletRepeat moveDown(ISmartletGroup group)`

Move given group down a row.

Parameters:

`group` -

Returns:

this

Since:

5.8.0

---

moveFirst

`ISmartletRepeat **moveFirst**(ISmartletGroup group)`

Move given group to the top of the repeat.

Parameters:

`group` -

Returns:

this

Since:

5.8.0

---

moveLast

`ISmartletRepeat **moveLast**(ISmartletGroup group)`

Move given group to the bottom of the repeat.

Parameters:

`group` -

Returns:

this

Since:

5.8.0

---

sortByColumn

`ISmartletRepeat **sortByColumn**(string columnName)`

Sorts groups of repeat alphanumerically with given field's string value.

Parameters:

`columnName` -

Returns:

this

Since:

5.8.0

---

sortByColumnDescending

`ISmartletRepeat **sortByColumnDescending**(string columnName)`

Sorts groups of repeat alphanumerically with given field's string value.
Descending order version.

Parameters:

`columnName` -

Returns:

this

Since:

5.8.0

---

filterByColumn

```
ISmartletRepeat filterByColumn(string columnName,
                               string subString,
                               bool ignoreCase)
```

Filter the repeat's groups, keeping only those containing given substring in given column.

Parameters:

`columnName` -

`subString` -

Returns:

this

Since:

5.8.0

---

setReadonly

```
ISmartletRepeat setReadonly(bool readonly)
```

Sets the "readonly" flag for all fields in all groups of this repeat.

Specified by:

`setReadonly` in interface `ISmartletField`

Parameters:

`readonly` - - readonly or not.

Returns:

this

Since:

5.8.0

---

getSelectedGroupIndexes

```
int[] getSelectedGroupIndexes()
```

Gets an integer array of the selected rows.

Returns:

int[] array of groups indices, 0-based

Since:

7.1.0

---

getUnSelectedGroupIndexes

```
int[] getUnSelectedGroupIndexes()
```

Gets an integer array of the non selected rows.

Returns:

int[] array of groups indices, 0-based

Since:

7.1.0

---

com.alphinat.sg5.widget.knowledge

Interface ISmartletKnowledgeEntry

```
public interface ISmartletKnowledgeEntry
```

`ISmartletKnowledgeEntry` is an interface representing a Smartlet knowledge entry.

Method Summary

`ISmartletKnowledgeEntry[]`	**`getEntries`**`()` Gets the sub entries.
`string`	**`getLabel`**`()` Gets the knowledge entry label.
`string`	**`getName`**`()` Gets the entry name.
`string`	**`getValue`**`()` Gets the knowledge entry value.
`bool`	**`isRepeat`**`()` Is the entry repeated or not.
`void`	**`resetEntries`**`()` Reset the entries to the original defined ones.
`void`	**`setEntries`**`(ISmartletKnowledgeEntry[] entries)` Sets the sub entries.

Method Detail

getName

```
string getName()
```

Gets the entry name.

Returns:
knowledge entry name

---

getLabel

```
string getLabel()
```

Gets the knowledge entry label.

Returns:
knowledge entry label

---

getValue

```
string getValue()
```

Gets the knowledge entry value.

Returns:
knowledge entry value

---

isRepeat

```
bool isRepeat()
```

Is the entry repeated or not.

Returns:
true if the entry is repeated

---

getEntries

```
ISmartletKnowledgeEntry[] getEntries()
```

Gets the sub entries.

Returns:

sub entries

---

setEntries

```
void setEntries(ISmartletKnowledgeEntry[] entries)
```

Sets the sub entries.

Parameters:

`entries` - - knowledge entries.

---

resetEntries

```
void resetEntries()
```

Reset the entries to the original defined ones.

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5.widget.knowledge

Interface ISmartletKnowledge

All Superinterfaces:

ISmartletElement, ISmartletField

`public interface` **`ISmartletKnowledge`**
extends ISmartletField

`ISmartletKnowledge` is an interface representing a Smartlet knowledge widget.

Method Summary

`ISmartletKnowledgeEntry[]`	**`getKnowledgeEntries`**`()` Gets the knowledge entries.
`void`	**`resetEntries`**`()` Reset the entries to their original state.
`void`	**`setKnowledgeEntries`**`(ISmartletKnowledgeEntry[] entries)` Sets the knowledge entries

Methods inherited from interface com.alphinat.sg5.ISmartletField

`addSourceField, addTargetField, addTargetField, addTargetFieldByName, appendAfter, appendBefore, appendTo, appendTo, applyDefinition, calculate, calculateAvailability, clear, detach, getChoiceLayout, getCSSClass, getCSSHeight, getCSSStyle, getCSSWidth, getErrorCodes, getErrorMessages, getEventSource, getEventTarget, getFormat, getHelp, getHelpId, getHtmlName, getId, getLabel, getLayoutAttribute, getLayoutAttributes, getMaxLength, getMetaData, getMetaDataNames, getMinLength, getName, getNext, getPage, getParent, getPlacement, getPrefix, getPrevious, getRepeatIndex, getRepeatSelectedStrings, getRepeatSelectedValues, getRepeatStrings, getRepeatValues,` getSmartlet, getString, getSuffix, getTooltip, getTypeConst, getTypeDetail, getValue, getValueParseScript, isAvailable, isEncrypted, isHelpLink, isPersistent, isReadonly, isRequired, isRequiredOnSummaryOnly, isShownInSummarySection, isUnderRepeatDefaultGroup, isValid, removeSourceField, removeTargetField, removeTargetField, removeTargetFieldByName, resetValidationDefinitions, resetValidationResult, setAvailabilityScript, setCalculationScript, setChoiceLayout, setCSSClass, setCSSHeight, setCSSStyle, setCSSWidth, setEncrypted, setError, setFormat, setHelp, setLabel, setLayoutAttributes, setMetaData, setPersistent, setPlacement, setPosition, setPrefix, setReadonly, setReadonly, setRepeatStrings, setRepeatValues, setString, setSuffix, setTooltip, setValidateOnSummaryOnly, setValidationRequire, setValidationScript, setValue, setValueParseScript, triggerEvent, validate

Methods inherited from interface com.alphinat.sg5.ISmartletElement

accept, data, data, getDataNames

Method Detail

getKnowledgeEntries

```
ISmartletKnowledgeEntry[] getKnowledgeEntries()
```

Gets the knowledge entries.

Returns:
ISmartletKnowledgeEntry

---

setKnowledgeEntries

```
void setKnowledgeEntries(ISmartletKnowledgeEntry[] entries)
```

Sets the knowledge entries

Parameters:
`entries` - - array of knowledge entries.

resetEntries

```
void resetEntries()
```

Reset the entries to their original state.

com.alphinat.sg5.widget.group

Interface ISmartletGroup

All Superinterfaces:

ISmartletElement, ISmartletField

`public interface` **`ISmartletGroup`**
extends ISmartletField

`ISmartletGroup` is an interface representing a Smartlet group.

Method Summary

`void`	**`clear`**`()` Clears subfields.
`ISmartletField[]`	**`findAllFields`**`()` Returns all fields under the group.
`ISmartletField`	**`findFieldById`**`(string id)` Finds the first matching field by id inside the group.
`ISmartletField`	**`findFieldByName`**`(string name)` Finds the first matching field by name inside the group.
`ISmartletField[]`	**`findFieldsByRegex`**`(string regularExpression)` Finds fields by regulation expression under the group.
`ISmartletField[]`	**`findFieldsByScript`**`(string script)` Finds fields by matching script under the group.
`ISmartletField[]`	**`findFieldsByTypes`**`(int[] types)` Finds fields by types under the group.
`ISmartletField[]`	**`getFields`**`()` Gets the top level fields of the group.
`ISmartletField[]`	**`getShownFields`**`()`

	Gets the available fields directly under the group.
`bool`	**`isGroupSelected`**`()` Verify is a group is selected, when in the context of a repeat field.
`void`	**`selectGroup`**`()` Select a group, when in the context of a repeat field.
`void`	**`unSelectGroup`**`()` Un-select a group, when in the context of a repeat field.

Methods inherited from interface com.alphinat.sg5.ISmartletField

addSourceField, addTargetField, addTargetField, addTargetFieldByName, appendAfter, appendBefore, appendTo, appendTo, applyDefinition, calculate, calculateAvailability, detach, getChoiceLayout, getCSSClass, getCSSHeight, getCSSStyle, getCSSWidth, getErrorCodes, getErrorMessages, getEventSource, getEventTarget, getFormat, getHelp, getHelpId, getHtmlName, getId, getLabel, getLayoutAttribute, getLayoutAttributes, getMaxLength, getMetaData, getMetaDataNames, getMinLength, getName, getNext, getPage, getParent, getPlacement, getPrefix, getPrevious, getRepeatIndex, getRepeatSelectedStrings, getRepeatSelectedValues, getRepeatStrings, getRepeatValues, getSmartlet, getString, getSuffix, getTooltip, getTypeConst, getTypeDetail, getValue, getValueParseScript, isAvailable, isEncrypted, isHelpLink, isPersistent, isReadonly, isRequired, isRequiredOnSummaryOnly, isShownInSummarySection, isUnderRepeatDefaultGroup, isValid, removeSourceField, removeTargetField, removeTargetField, removeTargetFieldByName, resetValidationDefinitions, resetValidationResult, setAvailabilityScript, setCalculationScript, setChoiceLayout, setCSSClass, setCSSHeight, setCSSStyle, setCSSWidth, setEncrypted, setError, setFormat, setHelp, setLabel, setLayoutAttributes, setMetaData, setPersistent, setPlacement, setPosition, setPrefix, setReadonly, setReadonly, setRepeatStrings, setRepeatValues, setString, setSuffix, setTooltip, setValidateOnSummaryOnly, setValidationRequire, setValidationScript, setValue, setValueParseScript, triggerEvent, validate

Methods inherited from interface com.alphinat.sg5.ISmartletElement

accept, data, data, getDataNames

Method Detail

getFields

`ISmartletField[] getFields()`

Gets the top level fields of the group.

Returns:

top level fields inside the group.

---

findFieldByName

`ISmartletField findFieldByName(string name)`

Finds the first matching field by name inside the group.

Parameters:

`name` - - Field name.

Returns:

`Smartlet field`

---

findFieldById

`ISmartletField findFieldById(string id)`

Finds the first matching field by id inside the group.

Parameters:

`id` -

Returns:

`Smartlet field`

---

findFieldsByTypes

`ISmartletField[] findFieldsByTypes(int[] types)`

Finds fields by types under the group.

Parameters:

`types` - - field types to find

Returns:

`array of Smartlet fields`

Since:

5.4.0

---

findFieldsByRegex

`ISmartletField[] findFieldsByRegex(string regularExpression)`

Finds fields by regulation expression under the group.

Parameters:

`regularExpression` - - regulation expression to match the field name

Returns:

`array of Smartlet fields`

Since:

5.4.0

---

findFieldsByScript

`ISmartletField[] findFieldsByScript(string script)`

Finds fields by matching script under the group.

Parameters:

`script` - - BSH script

Returns:

`array of Smartlet fields`

Since:

5.4.0

---

findAllFields

`ISmartletField[] findAllFields()`

Returns all fields under the group.

Returns:

`array of Smartlet fields`

---

getShownFields

```
ISmartletField[] getShownFields()
```

Gets the available fields directly under the group.

Returns:

array of `ISmartletField` on the page

Since:

5.4.0

---

clear

```
void clear()
```

Clears subfields.
Note: this method will not clean the uploaded files on the disk.

Specified by:

`clear` in interface `ISmartletField`

Since:

5.8.0

---

selectGroup

```
void selectGroup()
```

Select a group, when in the context of a repeat field.

Since:

7.1.0

---

unSelectGroup

```
void unSelectGroup()
```

Un-select a group, when in the context of a repeat field.

Since:

7.1.0

---

isGroupSelected

```
bool isGroupSelected()
```

Verify is a group is selected, when in the context of a repeat field.

Since:

7.1.0

---

com.alphinat.sg5.widget.date

Interface ISmartletDate

All Superinterfaces:

ISmartletElement, ISmartletField

`public interface` **`ISmartletDate`**
extends ISmartletField

`ISmartletDate` is an interface representing a Smartlet date field.

Method Summary

`int`	**`getDay`**`()` Gets day of month field, starting from 1.
`int`	**`getEndYear`**`()` Returns end year.
`int`	**`getMonth`**`()` Gets month of the date field, starting from 1.
`string`	**`getMonthString`**`()` Get month string of date field.
`int`	**`getStartYear`**`()` Returns starting year.
`ISmartletField[]`	**`getSubControls`**`()` Returns array of controls.
`int`	**`getYear`**`()` Gets year of the date field.
`bool`	**`isMultipleControls`**`()` Returns true if date field is shown as multiple controls.
`void`	**`setDay`**`(int day)`

	Sets day of date field.
`void`	**`setMonth`**`(int month)` Sets month of date field
`void`	**`setMonthString`**`(string month)` Set month string of date field.
`void`	**`setYear`**`(int year)` Sets year of date field.

Methods inherited from interface com.alphinat.sg5.ISmartletField

`addSourceField, addTargetField, addTargetField, addTargetFieldByName, appendAfter, appendBefore, appendTo, appendTo, applyDefinition, calculate, calculateAvailability, clear, detach, getChoiceLayout, getCSSClass, getCSSHeight, getCSSStyle, getCSSWidth, getErrorCodes, getErrorMessages, getEventSource, getEventTarget, getFormat, getHelp, getHelpId, getHtmlName, getId, getLabel, getLayoutAttribute, getLayoutAttributes, getMaxLength, getMetaData, getMetaDataNames, getMinLength, getName, getNext, getPage, getParent, getPlacement, getPrefix, getPrevious, getRepeatIndex, getRepeatSelectedStrings, getRepeatSelectedValues, getRepeatStrings, getRepeatValues, getSmartlet, getString, getSuffix, getTooltip, getTypeConst, getTypeDetail, getValue, getValueParseScript, isAvailable, isEncrypted, isHelpLink, isPersistent, isReadonly, isRequired, isRequiredOnSummaryOnly, isShownInSummarySection, isUnderRepeatDefaultGroup, isValid, removeSourceField, removeTargetField, removeTargetField, removeTargetFieldByName, resetValidationDefinitions, resetValidationResult, setAvailabilityScript, setCalculationScript, setChoiceLayout, setCSSClass, setCSSHeight, setCSSStyle, setCSSWidth, setEncrypted, setError, setFormat, setHelp, setLabel, setLayoutAttributes, setMetaData, setPersistent, setPlacement, setPosition, setPrefix, setReadonly, setReadonly, setRepeatStrings, setRepeatValues, setString, setSuffix, setTooltip, setValidateOnSummaryOnly, setValidationRequire, setValidationScript, setValue, setValueParseScript, triggerEvent, validate`

Methods inherited from interface com.alphinat.sg5.ISmartletElement

`accept, data, data, getDataNames`

Method Detail

isMultipleControls

```
bool isMultipleControls()
```

Returns true if date field is shown as multiple controls.

Returns:

bool

Since:

5.4.0

---

getSubControls

```
ISmartletField[] getSubControls()
```

Returns array of controls. If date field is shown as multiple controls, the array contains year control, month control, day control and separators. If date field is shown as one control, the arry only contain the ISmartletDate itself.

Returns:

array of controls

Since:

5.4.0

---

getStartYear

```
int getStartYear()
```

Returns starting year. 1900 is returned if starting year is not specified.

Returns:

Since:

5.4.0

---

getEndYear

```
int getEndYear()
```

Returns end year. 2100 is returned if end year is not specified.

Returns:

Since:

5.4.0

---

getYear

```
int getYear()
```

Gets year of the date field.

Returns:

year of date

---

setYear

```
void setYear(int year)
```

Sets year of date field.

Parameters:

`year` -

---

getMonth

```
int getMonth()
```

Gets month of the date field, starting from 1.

Returns:

month of date

---

setMonth

```
void setMonth(int month)
```

Sets month of date field

Parameters:

`month` -

---

getDay

```
int getDay()
```

Gets day of month field, starting from 1.

Returns:

day of month

---

setDay

```
void setDay(int day)
```

Sets day of date field.

Parameters:

`day` -

---

getMonthString

```
string getMonthString()
```

Get month string of date field.

Returns:

month string

---

setMonthString

```
void setMonthString(string month)
```

Set month string of date field.

Parameters:

`month` -

---

com.alphinat.sg5.widget.upload

Interface ISmartletUpload

All Superinterfaces:

ISmartletElement, ISmartletField

`public interface` **`ISmartletUpload`**
extends ISmartletField

`ISmartletUpload` is an interface representing a Smartlet upload field.

Since:

SGS V7.0.0

Method Summary

`void`	**`deleteFile`**`()` Deletes uploaded file.
`string`	**`getBase64EncodedValue`**`()` Return the base64 encoded file content
`byte[]`	**`getBytes`**`()` Gets file bytes.
`string`	**`getFileExtension`**`()` Return the file extension
`string`	**`getFileName`**`()` Gets uploaded file name.
`string`	**`getFilePath`**`()` Gets uploaded file path when in disk mode for upload files.
`long`	**`getFileSize`**`()` Get size of uploaded file.
`string`	

	getMimeType() Return the mime type for the file based on the extension
bool	**isFileEmpty**() Return true if file is empty

Methods inherited from interface com.alphinat.sg5.ISmartletField

addSourceField, addTargetField, addTargetField, addTargetFieldByName, appendAfter, appendBefore, appendTo, appendTo, applyDefinition, calculate, calculateAvailability, clear, detach, getChoiceLayout, getCSSClass, getCSSHeight, getCSSStyle, getCSSWidth, getErrorCodes, getErrorMessages, getEventSource, getEventTarget, getFormat, getHelp, getHelpId, getHtmlName, getId, getLabel, getLayoutAttribute, getLayoutAttributes, getMaxLength, getMetaData, getMetaDataNames, getMinLength, getName, getNext, getPage, getParent, getPlacement, getPrefix, getPrevious, getRepeatIndex, getRepeatSelectedStrings, getRepeatSelectedValues, getRepeatStrings, getRepeatValues, getSmartlet, getString, getSuffix, getTooltip, getTypeConst, getTypeDetail, getValue, getValueParseScript, isAvailable, isEncrypted, isHelpLink, isPersistent, isReadonly, isRequired, isRequiredOnSummaryOnly, isShownInSummarySection, isUnderRepeatDefaultGroup, isValid, removeSourceField, removeTargetField, removeTargetField, removeTargetFieldByName, resetValidationDefinitions, resetValidationResult, setAvailabilityScript, setCalculationScript, setChoiceLayout, setCSSClass, setCSSHeight, setCSSStyle, setCSSWidth, setEncrypted, setError, setFormat, setHelp, setLabel, setLayoutAttributes, setMetaData, setPersistent, setPlacement, setPosition, setPrefix, setReadonly, setReadonly, setRepeatStrings, setRepeatValues, setString, setSuffix, setTooltip, setValidateOnSummaryOnly, setValidationRequire, setValidationScript, setValue, setValueParseScript, triggerEvent, validate

Methods inherited from interface com.alphinat.sg5.ISmartletElement

accept, data, data, getDataNames

Method Detail

getFileName

`string getFileName()`

Gets uploaded file name. Returns empty string if no uploaded file.

Returns:

file name

---

getFilePath

`string getFilePath()`

Gets uploaded file path when in disk mode for upload files. Returns empty string if using memory mode.

Returns:

file name

---

getFileSize

`long getFileSize()`

Get size of uploaded file. Returns 0 if no uploaded file.

Returns:

long value

---

getBytes

`byte[] getBytes()`

Gets file bytes. Returns null if no uploaded file.

Returns:

bytes

---

deleteFile

`void deleteFile()`

Deletes uploaded file. If file is stored in file system. The file is deleted.

---

isFileEmpty

```
bool isFileEmpty()
```

Return true if file is empty

Returns:

bool

---

getMimeType

```
string getMimeType()
```

Return the mime type for the file based on the extension

Returns:

string

Since:

V7.1.0

---

getBase64EncodedValue

```
string getBase64EncodedValue()
```

Return the base64 encoded file content

Returns:

string

Since:

V7.1.0

---

getFileExtension

```
string getFileExtension()
```

Return the file extension

Returns:

string

Since:

V7.1.0

---

com.alphinat.sg5

Interface Constants

```
public interface Constants
```

Nested Class Summary

`static interface`	**`Constants.ElementType`** Smartlet element type constant.
`static interface`	**`Constants.EmailFormat`** Email formats supported for sendmail API
`static interface`	**`Constants.ErrorCode`** Error Code constant.
`static interface`	**`Constants.FileType`** Smartlet file type constant.
`static interface`	**`Constants.Scope`**
`static interface`	**`Constants.SmartletEvent`** Smartlet event type constant.
`static interface`	**`Constants.WSInputBehavior`** Dynamic input behavior on null.

com.alphinat.sg5

Interface Constants.SmartletEvent

Enclosing interface:
Constants

```
public static interface Constants.SmartletEvent
```

Smartlet event type constant.

Field Summary

`static int`	**EVENT_BLUR** Field events - on blur
`static int`	**EVENT_BUTTON_CLICK** Event triggered when click button.
`static int`	**EVENT_BUTTON_DBLCLICK** Event triggered when double click button.
`static int`	**EVENT_CHANGE** Field events - on change
`static int`	**EVENT_CLICK** Field events - on click
`static int`	**EVENT_FIELD_INIT** Field events - on field initialization
`static int`	**EVENT_FIELD_RENDER** Field events - on rendering field
`static int`	**EVENT_FOCUS** Field events - on focus
`static int`	**EVENT_INPUT** Field events - on input

`static int`	**EVENT_KEYDOWN** Field events - on key down
`static int`	**EVENT_KEYPRESS** Field events - on key press
`static int`	**EVENT_KEYUP** Field events - on key up
`static int`	**EVENT_MOUSEOUT** Event triggered when double click button.
`static int`	**EVENT_MOUSEOVER** Event triggered when double click button.
`static int`	**EVENT_ON_ENTER_PAGE** Event triggered on entering a smartlet page.
`static int`	**EVENT_ON_EXIT_PAGE** Event triggered on leaving a smartlet page.
`static int`	**EVENT_ON_INIT_SMARTLET** Smartlet initialization event.
`static int`	**EVENT_PAGE_INIT** Page events - on page initialization
`static int`	**EVENT_PAGE_RENDER** Page events - on rendering page
`static int`	**EVENT_SELECT** Field events - on select

Field Detail

EVENT_ON_INIT_SMARTLET

```
static final int EVENT_ON_INIT_SMARTLET
```

Smartlet initialization event.

See Also:

Constant Field Values

EVENT_ON_ENTER_PAGE

```
static final int EVENT_ON_ENTER_PAGE
```

Event triggered on entering a smartlet page.

See Also:

Constant Field Values

EVENT_ON_EXIT_PAGE

```
static final int EVENT_ON_EXIT_PAGE
```

Event triggered on leaving a smartlet page.

See Also:

Constant Field Values

EVENT_BUTTON_CLICK

```
static final int EVENT_BUTTON_CLICK
```

Event triggered when click button.

See Also:

Constant Field Values

EVENT_CLICK

```
static final int EVENT_CLICK
```

Field events - on click

Since:

7.0.0

See Also:

Constant Field Values

EVENT_BLUR

```
static final int EVENT_BLUR
```

Field events - on blur

Since:

7.0.0

See Also:

Constant Field Values

EVENT_CHANGE

```
static final int EVENT_CHANGE
```

Field events - on change

Since:

7.0.0

See Also:

Constant Field Values

EVENT_FOCUS

```
static final int EVENT_FOCUS
```

Field events - on focus

Since:

7.0.0

See Also:

Constant Field Values

EVENT_INPUT

```
static final int EVENT_INPUT
```

Field events - on input

Since:

7.0.0

See Also:

Constant Field Values

EVENT_KEYDOWN

```
static final int EVENT_KEYDOWN
```

Field events - on key down

Since:

7.0.0

See Also:

Constant Field Values

EVENT_KEYPRESS

```
static final int EVENT_KEYPRESS
```

Field events - on key press

Since:

7.0.0

See Also:

Constant Field Values

EVENT_KEYUP

```
static final int EVENT_KEYUP
```

Field events - on key up

Since:

7.0.0

See Also:

Constant Field Values

EVENT_SELECT

```
static final int EVENT_SELECT
```

Field events - on select

Since:
7.0.0
See Also:
Constant Field Values

EVENT_FIELD_RENDER

```
static final int EVENT_FIELD_RENDER
```

Field events - on rendering field

Since:
7.0.0
See Also:
Constant Field Values

EVENT_FIELD_INIT

```
static final int EVENT_FIELD_INIT
```

Field events - on field initialization

Since:
7.0.0
See Also:
Constant Field Values

EVENT_PAGE_RENDER

```
static final int EVENT_PAGE_RENDER
```

Page events - on rendering page

Since:

7.0.0

See Also:

Constant Field Values

EVENT_PAGE_INIT

```
static final int EVENT_PAGE_INIT
```

Page events - on page initialization

Since:

7.0.0

See Also:

Constant Field Values

EVENT_BUTTON_DBLCLICK

```
static final int EVENT_BUTTON_DBLCLICK
```

Event triggered when double click button.

See Also:

Constant Field Values

EVENT_MOUSEOVER

```
static final int EVENT_MOUSEOVER
```

Event triggered when double click button.

See Also:

Constant Field Values

EVENT_MOUSEOUT

```
static final int EVENT_MOUSEOUT
```

Event triggered when double click button.

See Also:

Constant Field Values

com.alphinat.sg5

Interface Constants.Scope

Enclosing interface:

Constants

```
public static interface Constants.Scope
```

Field Summary

`static int`	**APPLICATION** The environment scope whose `string` representation is: `application`.
`static int`	**CONFIGURATION** The environment scope whose `string` representation is: `configuration`.
`static int`	**PARAMETER** The environment scope whose `string` representation is: `parameter`.
`static int`	**PREFERENCE** The environment scope whose `string` representation is: `portlet_preference`.
`static int`	**REQUEST** The environment scope whose `string` representation is: `request`.
`static int`	**SESSION** The environment scope whose `string` representation is: `session`.

Field Detail

REQUEST

`static final int` **`REQUEST`**

The environment scope whose `string` representation is: `request`.

See Also:
Constant Field Values

SESSION

`static final int` **`SESSION`**

The environment scope whose `string` representation is: `session`.

See Also:
Constant Field Values

APPLICATION

`static final int` **`APPLICATION`**

The environment scope whose `string` representation is: `application`.

See Also:
Constant Field Values

PREFERENCE

`static final int` **`PREFERENCE`**

The environment scope whose `string` representation is: `portlet_preference`.

See Also:
Constant Field Values

PARAMETER

```
static final int PARAMETER
```

The environment scope whose `string` representation is: `parameter`. Each value within the scope is wrapped in a `ArrayList`.

See Also:

Constant Field Values

CONFIGURATION

```
static final int CONFIGURATION
```

The environment scope whose `string` representation is: `configuration`.

See Also:

Constant Field Values

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5

Interface Constants.FileType

Enclosing interface:

Constants

```
public static interface Constants.FileType
```

Smartlet file type constant.

Field Summary

`static int`	**`PDF`** PDF file
`static int`	**`XML`** XML file

Field Detail

PDF

```
static final int PDF
```

PDF file

See Also:

Constant Field Values

XML

```
static final int XML
```

XML file

See Also:

Constant Field Values

com.alphinat.sg5

Interface Constants.ErrorCode

Enclosing interface:

Constants

`public static interface` **`Constants.ErrorCode`**

Error Code constant.

Field Summary

`static int`	**`Error_Adv_Rule`** Rule based validation error.
`static int`	**`Error_Adv_Script`** Advanced script validation error.
`static int`	**`Error_Date_Format`** Date field format error.
`static int`	**`Error_Date_Invalid`** Invalid date.
`static int`	**`Error_Ext_Validation`** External validation error.
`static int`	**`Error_FileType_Format`** File type field format error.
`static int`	**`Error_Format`** Field format error.
`static int`	**`Error_Goto_Summary`** Cannot goto summary section because of changed branching.
`static int`	**`Error_Mandatory`** Mandatory field validation error.

`static int`	**`Error_Maxlength`** Field maximum length validation error.
`static int`	**`Error_Minlength`** Field minimum length validation error.
`static int`	**`Error_Number_Format`** Number field format error.
`static int`	**`Error_Other`** Unclassified error.
`static int`	**`Error_Regexp_Validation`** Field format regular expression validation error.

Field Detail

Error_Mandatory

```
static final int Error_Mandatory
```

Mandatory field validation error.

See Also:

Constant Field Values

Error_Minlength

```
static final int Error_Minlength
```

Field minimum length validation error.

See Also:

Constant Field Values

Error_Maxlength

```
static final int Error_Maxlength
```

Field maximum length validation error.

See Also:

Constant Field Values

Error_Regexp_Validation

```
static final int Error_Regexp_Validation
```

Field format regular expression validation error.

See Also:

Constant Field Values

Error_Number_Format

```
static final int Error_Number_Format
```

Number field format error.

See Also:

Constant Field Values

Error_Date_Format

```
static final int Error_Date_Format
```

Date field format error.

See Also:

Constant Field Values

Error_Format

```
static final int Error_Format
```

Field format error.

Since:

5.4.0

See Also:

Constant Field Values

Error_Date_Invalid

```
static final int Error_Date_Invalid
```

Invalid date.

See Also:

Constant Field Values

Error_FileType_Format

```
static final int Error_FileType_Format
```

File type field format error.

See Also:

Constant Field Values

Error_Adv_Rule

```
static final int Error_Adv_Rule
```

Rule based validation error.

See Also:

Constant Field Values

Error_Adv_Script

```
static final int Error_Adv_Script
```

Advanced script validation error.

See Also:

Constant Field Values

Error_Ext_Validation

```
static final int Error_Ext_Validation
```

External validation error.

See Also:

Constant Field Values

Error_Goto_Summary

```
static final int Error_Goto_Summary
```

Cannot goto summary section because of changed branching.

See Also:

Constant Field Values

Error_Other

```
static final int Error_Other
```

Unclassified error.

See Also:

Constant Field Values

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview Package Class Use Tree Deprecated Index Help
PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

com.alphinat.sg5

Interface Constants.ElementType

Enclosing interface:

Constants

```
public static interface Constants.ElementType
```

Smartlet element type constant.

Field Summary

`static int`	**BUTTON** The element is a button field.
`static string`	**BUTTON_GLOBAL_NAVIGATION** The element is a global navigation button.
`static string`	**BUTTON_GOTO_SUMMARY** The element is a button to navigate to summary page.
`static string`	**BUTTON_MODIFY_PAGE** The element is a button to modify a page from summary section.
`static string`	**BUTTON_NEXT_PAGE** The element is a button to navigate to next page.
`static string`	**BUTTON_PREVIOUS_PAGE** The element is a button to navigate to previous page.
`static string`	**BUTTON_REFRESH_PAGE** The element is a button to refresh current page.
`static string`	**BUTTON_REPEAT_DELETE** The element is a button to remove an instance for repeat.
`static string`	**BUTTON_REPEAT_INSERT** The element is a button to insert an instance for repeat.

static string	**BUTTON_SUBSMARTLET_ENTER** The element is a button to enter sub smartlet.
static string	**BUTTON_SUBSMARTLET_RETURN_SAVE** The element is a button to return from sub smartlet.
static string	**BUTTON_SUBSMARTLET_WITHOUT_SAVE** The element is a button to return from sub smartlet without save.
static int	**CHECK** The element is a checkbox field.
static int	**COL** The element is a smartlet service.
static int	**DATE** The element is a date field.
static int	**DROP** The element is a dropdown field.
static int	**GROUP** The element is a group widget.
static int	**HIDDEN** The element is a hidden field.
static int	**KNOWLEDGE** The element is a knowledge widget.
static int	**KNOWLEDGE_ENTRY** The element is a knowledge entry of a knowledge widget.
static int	**LBOX** The element is a listbox field.
static int	**NUMBER** The element is a number field.
static int	**OPTION** The element is an option.
static int	**OPTION_GROUP** The element is an option group.
static int	**PAGE** The element is a smartlet page.

`static int`	**PASSWORD** The element is a password field.
`static int`	**RADIO** The element is a radio button field.
`static int`	**REPEAT** The element is a repeat widget.
`static int`	**ROW** The element is a smartlet service.
`static int`	**SERVICE** The element is a smartlet service.
`static int`	**SMARTLET** The element is a smartlet.
`static int`	**STATIC_IMG** The element is a static image field.
`static int`	**STATIC_TEXT** The element is a static text field.
`static int`	**SUB_SMARTLET** The element is a subsmartlet widget.
`static int`	**SUMMARY** The element is a summary section.
`static int`	**TEXT** The element is a text field.
`static int`	**TEXTAREA** The element is a textarea field.
`static int`	**UPLOAD** The element is an upload field.

Field Detail

SMARTLET

```
static final int SMARTLET
```

The element is a smartlet.

See Also:

Constant Field Values

PAGE

```
static final int PAGE
```

The element is a smartlet page.

See Also:

Constant Field Values

GROUP

```
static final int GROUP
```

The element is a group widget.

See Also:

Constant Field Values

REPEAT

```
static final int REPEAT
```

The element is a repeat widget.

See Also:

Constant Field Values

TEXT

```
static final int TEXT
```

The element is a text field.

See Also:
Constant Field Values

TEXTAREA

```
static final int TEXTAREA
```

The element is a textarea field.

See Also:
Constant Field Values

PASSWORD

```
static final int PASSWORD
```

The element is a password field.

See Also:
Constant Field Values

HIDDEN

```
static final int HIDDEN
```

The element is a hidden field.

See Also:
Constant Field Values

DATE

```
static final int DATE
```

The element is a date field.

See Also:

Constant Field Values

NUMBER

```
static final int NUMBER
```

The element is a number field.

See Also:

Constant Field Values

CHECK

```
static final int CHECK
```

The element is a checkbox field.

See Also:

Constant Field Values

DROP

```
static final int DROP
```

The element is a dropdown field.

See Also:

Constant Field Values

LBOX

`static final int **LBOX**`

The element is a listbox field.

See Also:
Constant Field Values

RADIO

`static final int **RADIO**`

The element is a radio button field.

See Also:
Constant Field Values

OPTION

`static final int **OPTION**`

The element is an option.

See Also:
Constant Field Values

OPTION_GROUP

`static final int **OPTION_GROUP**`

The element is an option group.

See Also:
Constant Field Values

STATIC_TEXT

```
static final int STATIC_TEXT
```

The element is a static text field.

See Also:

Constant Field Values

STATIC_IMG

```
static final int STATIC_IMG
```

The element is a static image field.

See Also:

Constant Field Values

UPLOAD

```
static final int UPLOAD
```

The element is an upload field.

See Also:

Constant Field Values

BUTTON

```
static final int BUTTON
```

The element is a button field.

See Also:

Constant Field Values

KNOWLEDGE

```
static final int KNOWLEDGE
```

The element is a knowledge widget.

See Also:

Constant Field Values

KNOWLEDGE_ENTRY

```
static final int KNOWLEDGE_ENTRY
```

The element is a knowledge entry of a knowledge widget.

See Also:

Constant Field Values

SUB_SMARTLET

```
static final int SUB_SMARTLET
```

The element is a subsmartlet widget.

See Also:

Constant Field Values

SUMMARY

```
static final int SUMMARY
```

The element is a summary section.

See Also:

Constant Field Values

SERVICE

```
static final int SERVICE
```

The element is a smartlet service.

See Also:
Constant Field Values

ROW

```
static final int ROW
```

The element is a smartlet service.

See Also:
Constant Field Values

COL

```
static final int COL
```

The element is a smartlet service.

See Also:
Constant Field Values

BUTTON_SUBSMARTLET_ENTER

```
static final string BUTTON_SUBSMARTLET_ENTER
```

The element is a button to enter sub smartlet.

See Also:
Constant Field Values

BUTTON_SUBSMARTLET_RETURN_SAVE

`static final string` **`BUTTON_SUBSMARTLET_RETURN_SAVE`**

The element is a button to return from sub smartlet.

See Also:
Constant Field Values

BUTTON_SUBSMARTLET_WITHOUT_SAVE

`static final string` **`BUTTON_SUBSMARTLET_WITHOUT_SAVE`**

The element is a button to return from sub smartlet without save.

See Also:
Constant Field Values

BUTTON_NEXT_PAGE

`static final string` **`BUTTON_NEXT_PAGE`**

The element is a button to navigate to next page.

See Also:
Constant Field Values

BUTTON_PREVIOUS_PAGE

`static final string` **`BUTTON_PREVIOUS_PAGE`**

The element is a button to navigate to previous page.

See Also:
Constant Field Values

BUTTON_REPEAT_INSERT

`static final string` **`BUTTON_REPEAT_INSERT`**

The element is a button to insert an instance for repeat.

See Also:
Constant Field Values

BUTTON_REPEAT_DELETE

`static final string` **`BUTTON_REPEAT_DELETE`**

The element is a button to remove an instance for repeat.

See Also:
Constant Field Values

BUTTON_GOTO_SUMMARY

`static final string` **`BUTTON_GOTO_SUMMARY`**

The element is a button to navigate to summary page.

See Also:
Constant Field Values

BUTTON_MODIFY_PAGE

`static final string` **`BUTTON_MODIFY_PAGE`**

The element is a button to modify a page from summary section.

See Also:
Constant Field Values

BUTTON_REFRESH_PAGE

```
static final string BUTTON_REFRESH_PAGE
```

The element is a button to refresh current page.

See Also:

Constant Field Values

BUTTON_GLOBAL_NAVIGATION

```
static final string BUTTON_GLOBAL_NAVIGATION
```

The element is a global navigation button.

See Also:

Constant Field Values

---

com.alphinat.sg5

Interface Constants.EmailFormat

Enclosing interface:

Constants

```
public static interface Constants.EmailFormat
```

Email formats supported for sendmail API

Field Summary

`static int`	**DEFAULT**
`static int`	**HTMLONLY** Sends as HTML only.
`static int`	**TEXTHTML** Sends as text and HTML.
`static int`	**TEXTONLY** Sends as text only.

Field Detail

DEFAULT

```
static final int DEFAULT
```

See Also:

Constant Field Values

TEXTHTML

```
static final int TEXTHTML
```

Sends as text and HTML. This is the default.

See Also:
Constant Field Values

TEXTONLY

```
static final int TEXTONLY
```

Sends as text only.

See Also:
Constant Field Values

HTMLONLY

```
static final int HTMLONLY
```

Sends as HTML only.

See Also:
Constant Field Values

---

com.alphinat.sg5

Interface Constants.WSInputBehavior

Enclosing interface:

Constants

```
public static interface Constants.WSInputBehavior
```

Dynamic input behavior on null.

Field Summary

`static int`	**`DEFAULT`** Default, which trims optional node that are empty or null.
`static int`	**`EMPTY`** Empty, which means optional nodes returning empty string will be sent, while null will be trimmed.
`static int`	**`NULL`** Null, which means the node will be sent.

Field Detail

DEFAULT

```
static final int DEFAULT
```

Default, which trims optional node that are empty or null. For attributes the default is to keep it even if it is empty or null.

See Also:

Constant Field Values

EMPTY

```
static final int EMPTY
```

Empty, which means optional nodes returning empty string will be sent, while null will be trimmed. For attributes, means we don't keep it if the mapped value is null or empty.

See Also:
Constant Field Values

NULL

```
static final int NULL
```

Null, which means the node will be sent. For attributes, means we don't keep it if the mapped value is null.

See Also:
Constant Field Values

Constant Field Values

Contents

com.alphinat.*

com.alphinat.sg5.Constants.ElementType		
int	BUTTON	190000
string	BUTTON_GLOBAL_NAVIGATION	"button_global_navigation"
string	BUTTON_GOTO_SUMMARY	"button_goto_summary"
string	BUTTON_MODIFY_PAGE	"button_modify_page"
string	BUTTON_NEXT_PAGE	"button_next_page"
string	BUTTON_PREVIOUS_PAGE	"button_previous_page"
string	BUTTON_REFRESH_PAGE	"button_refresh_page"
string	BUTTON_REPEAT_DELETE	"button_repeat_delete"
string	BUTTON_REPEAT_INSERT	"button_repeat_insert"
string	BUTTON_SUBSMARTLET_ENTER	"button_subsmartlet_enter"
string	BUTTON_SUBSMARTLET_RETURN_SAVE	"button_subsmartlet_return_save"
string	BUTTON_SUBSMARTLET_WITHOUT_SAVE	"button_subsmartlet_without_save"
int	CHECK	110000
int	COL	250000
int	DATE	90000
int	DROP	120000
int	GROUP	30000
int	HIDDEN	80000
int	KNOWLEDGE	200000
int	KNOWLEDGE_ENTRY	201000
int	LBOX	130000

int	NUMBER	100000
int	OPTION	150000
int	OPTION_GROUP	151000
int	PAGE	20000
int	PASSWORD	70000
int	RADIO	140000
int	REPEAT	40000
int	ROW	240000
int	SERVICE	230000
int	SMARTLET	10000
int	STATIC_IMG	170000
int	STATIC_TEXT	160000
int	SUB_SMARTLET	210000
int	SUMMARY	220000
int	TEXT	50000
int	TEXTAREA	60000
int	UPLOAD	180000

com.alphinat.sg5.Constants.EmailFormat		
int	DEFAULT	1
int	HTMLONLY	3
int	TEXTHTML	1
int	TEXTONLY	2

com.alphinat.sg5.Constants.ErrorCode		
int	Error_Adv_Rule	-1301
int	Error_Adv_Script	-1302
int	Error_Date_Format	-1203
int	Error_Date_Invalid	-1204
int	Error_Ext_Validation	-1401
int	Error_FileType_Format	-1206
int	Error_Format	-1205
int	Error_Goto_Summary	-1501

int	Error_Mandatory	-1001
int	Error_Maxlength	-1102
int	Error_Minlength	-1101
int	Error_Number_Format	-1202
int	Error_Other	-9000
int	Error_Regexp_Validation	-1201

com.alphinat.sg5.Constants.FileType		
int	PDF	1
int	XML	2

com.alphinat.sg5.Constants.Scope		
int	APPLICATION	4
int	CONFIGURATION	8
int	PARAMETER	7
int	PREFERENCE	6
int	REQUEST	2
int	SESSION	3

com.alphinat.sg5.Constants.SmartletEvent		
int	EVENT_BLUR	9
int	EVENT_BUTTON_CLICK	4
int	EVENT_BUTTON_DBLCLICK	21
int	EVENT_CHANGE	10
int	EVENT_CLICK	4
int	EVENT_FIELD_INIT	18
int	EVENT_FIELD_RENDER	17
int	EVENT_FOCUS	11
int	EVENT_INPUT	12
int	EVENT_KEYDOWN	13
int	EVENT_KEYPRESS	14
int	EVENT_KEYUP	15
int	EVENT_MOUSEOUT	23

int	EVENT_MOUSEOVER	22
int	EVENT_ON_ENTER_PAGE	2
int	EVENT_ON_EXIT_PAGE	3
int	EVENT_ON_INIT_SMARTLET	1
int	EVENT_PAGE_INIT	20
int	EVENT_PAGE_RENDER	19
int	EVENT_SELECT	16

com.alphinat.sg5.Constants.WSInputBehavior		
int	DEFAULT	0
int	EMPTY	1
int	NULL	2

A

accept(ISmartletElementVisitor) - Method in interface com.alphinat.sg5.ISmartletElement

Implements the visitor's pattern to traverse Smartlet/page/field/services

add(Object) - Method in interface com.alphinat.sg5.widget.select.ISelectOptionList

Appends the specified element to the end of this list.

add(int, Object) - Method in interface com.alphinat.sg5.widget.select.ISelectOptionList

Inserts the specified element at the specified position in this list.

addActionError(Object, string, string) - Method in interface com.alphinat.sg5.ISmartlet

Adds an action error element composed of a source object, the error message and a stack trace.

addActionError(ISmartletActionError) - Method in interface com.alphinat.sg5.ISmartlet

Adds an action error object.

addGroup() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Creates an empty group and add it to the end.

addGroup(int) - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Create an empty group and insert it to the given position.

addGroup(ISmartletGroup) - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Adds a repeat group to the end.

addGroup(int, ISmartletGroup) - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Adds a repeat group to the specified position.

addLocalizedResource(String, string, string) - Method in interface com.alphinat.sg5.ISmartlet

Add a key/value pair to the translation resources

addPageToHistory(ISmartletPage) - Method in interface com.alphinat.sg5.ISmartlet

Adds given page to the history.

addSourceField(String) - Method in interface com.alphinat.sg5.ISmartletField

Add a source field to the current field, for autorefresh purposes.

addTargetField() - Method in interface com.alphinat.sg5.ISmartletField

Add the current field as a target to itself, for autorefresh purposes.

addTargetField(String) - Method in interface com.alphinat.sg5.ISmartletField

Add a target field to the current field, for autorefresh purposes.

addTargetFieldByName(String) - Method in interface com.alphinat.sg5.ISmartletField

Add a target field to the current field, for autorefresh purposes.

appendAfter(ISmartletField) - Method in interface com.alphinat.sg5.ISmartletField

Appends the field after given field.

appendBefore(ISmartletField) - Method in interface com.alphinat.sg5.ISmartletField

BUTTON_SUBSMARTLET_WITHOUT_SAVE - Static variable in interface com.alphinat.sg5.Constants.ElementType
The element is a button to return from sub smartlet without save.

C

calculate() - Method in interface com.alphinat.sg5.ISmartlet
Recalculates the page.
calculate() - Method in interface com.alphinat.sg5.ISmartletField
Recalculates the field value.
calculate() - Method in interface com.alphinat.sg5.ISmartletPage
Recalculates the page.
calculateAvailability() - Method in interface com.alphinat.sg5.ISmartletField
Calculates and returns the field availability.
call() - Method in interface com.alphinat.sg5.ISmartletService
Calls the service.
call(Object[]) - Method in interface com.alphinat.sg5.ISmartletService
Call the service with the provided parameters.
CHECK - Static variable in interface com.alphinat.sg5.Constants.ElementType
The element is a checkbox field.
clear() - Method in interface com.alphinat.sg5.ISmartlet
Calls clear on every page of this smartlet, recursively.
clear() - Method in interface com.alphinat.sg5.ISmartletField
Clears the fields.
clear() - Method in interface com.alphinat.sg5.ISmartletPage
Calls clear on every field of this page, recursively.
clear() - Method in interface com.alphinat.sg5.widget.group.ISmartletGroup
Clears subfields.
clear() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat
Clears the repeat instances, leaves one empty instance.
clear(Boolean) - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat
Deprecated. *since 6.5.0*
clear() - Method in interface com.alphinat.sg5.widget.select.ISelectOptionList
Removes all of the elements from this selection item list.
clearActionErrors() - Method in interface com.alphinat.sg5.ISmartlet
Clears all action error of this smartlet.
COL - Static variable in interface com.alphinat.sg5.Constants.ElementType
The element is a smartlet service.
com.alphinat.sg5 - package com.alphinat.sg5

com.alphinat.sg5.widget.date - package com.alphinat.sg5.widget.date

com.alphinat.sg5.widget.group - package com.alphinat.sg5.widget.group

com.alphinat.sg5.widget.knowledge - package com.alphinat.sg5.widget.knowledge

com.alphinat.sg5.widget.repeat - package com.alphinat.sg5.widget.repeat

com.alphinat.sg5.widget.select - package com.alphinat.sg5.widget.select

com.alphinat.sg5.widget.subsmartlet - package com.alphinat.sg5.widget.subsmartlet

com.alphinat.sg5.widget.summary - package com.alphinat.sg5.widget.summary

com.alphinat.sg5.widget.upload - package com.alphinat.sg5.widget.upload

CONFIGURATION - Static variable in interface com.alphinat.sg5.Constants.Scope
The environment scope whose `string` representation is: `configuration`.

Constants - Interface in com.alphinat.sg5

Constants.ElementType - Interface in com.alphinat.sg5
Smartlet element type constant.

Constants.EmailFormat - Interface in com.alphinat.sg5
Email formats supported for sendmail API

Constants.ErrorCode - Interface in com.alphinat.sg5
Error Code constant.

Constants.FileType - Interface in com.alphinat.sg5
Smartlet file type constant.

Constants.Scope - Interface in com.alphinat.sg5

Constants.SmartletEvent - Interface in com.alphinat.sg5
Smartlet event type constant.

Constants.WSInputBehavior - Interface in com.alphinat.sg5
Dynamic input behavior on null.

createDetachedGroup() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat
Creates a repeatable group instance.

createField(String, int) - Method in interface com.alphinat.sg5.ISmartlet
Create a dynamic field, a unique ID will be assigned to the new field.

createField(String, string, int) - Method in interface com.alphinat.sg5.ISmartlet
Create a dynamic field.

createField(ISmartletField) - Method in interface com.alphinat.sg5.ISmartlet
Create a dynamic field from existing field, a unique ID will be assigned to the new field.

createField(String, ISmartletField) - Method in interface com.alphinat.sg5.ISmartlet
Create a dynamic field from existing field.

createOption() - Method in interface com.alphinat.sg5.widget.select.ISelectOptionList

Error_Adv_Script - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Advanced script validation error.

Error_Date_Format - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Date field format error.

Error_Date_Invalid - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Invalid date.

Error_Ext_Validation - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
External validation error.

Error_FileType_Format - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
File type field format error.

Error_Format - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Field format error.

Error_Goto_Summary - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Cannot goto summary section because of changed branching.

Error_Mandatory - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Mandatory field validation error.

Error_Maxlength - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Field maximum length validation error.

Error_Minlength - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Field minimum length validation error.

Error_Number_Format - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Number field format error.

Error_Other - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Unclassified error.

Error_Regexp_Validation - Static variable in interface com.alphinat.sg5.Constants.ErrorCode
Field format regular expression validation error.

evalBSH(String) - Method in interface com.alphinat.sg5.ISmartlet
Evaluate Beanshell scripts.

EVENT_BLUR - Static variable in interface com.alphinat.sg5.Constants.SmartletEvent
Field events - on blur

EVENT_BUTTON_CLICK - Static variable in interface com.alphinat.sg5.Constants.SmartletEvent
Event triggered when click button.

EVENT_BUTTON_DBLCLICK - Static variable in interface com.alphinat.sg5.Constants.SmartletEvent
Event triggered when double click button.

EVENT_CHANGE - Static variable in interface com.alphinat.sg5.Constants.SmartletEvent
Field events - on change

EVENT_CLICK - Static variable in interface com.alphinat.sg5.Constants.SmartletEvent
Field events - on click

EVENT_FIELD_INIT - Static variable in interface com.alphinat.sg5.Constants.SmartletEvent
Field events - on field initialization

EVENT_FIELD_RENDER - Static variable in interface com.alphinat.sg5.Constants.SmartletEvent
Field events - on rendering field

getCSSStyle() - Method in interface com.alphinat.sg5.ISmartletPage

Gets the css style.

getCSSWidth() - Method in interface com.alphinat.sg5.ISmartletField

Gets the css width.

getCurrentLocale() - Method in interface com.alphinat.sg5.ISmartlet

Gets the current locale for the Smartlet

getCurrentLocaleDescription() - Method in interface com.alphinat.sg5.ISmartlet

Gets the current language for the Smartlet

getCurrentPage() - Method in interface com.alphinat.sg5.ISmartlet

Gets the current `page`.

getCurrentSmartlet() - Method in interface com.alphinat.sg5.ISmartlet

Gets the current `Smartlet`.

getDataNames() - Method in interface com.alphinat.sg5.ISmartletElement

Returns names of data stored.

getDay() - Method in interface com.alphinat.sg5.widget.date.ISmartletDate

Gets day of month field, starting from 1.

getDefaultGroup() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Returns default group as template.

getDomain() - Method in interface com.alphinat.sg5.ISmartlet

Gets current domain name

getEndYear() - Method in interface com.alphinat.sg5.widget.date.ISmartletDate

Returns end year.

getEnterButton() - Method in interface com.alphinat.sg5.widget.subsmartlet.ISubSmartletField

Gets a button to enter the subSmartlet.

getEntries() - Method in interface com.alphinat.sg5.widget.knowledge.ISmartletKnowledgeEntry

Gets the sub entries.

getEnvironment() - Method in interface com.alphinat.sg5.IServiceContext

Gets the Smartlet process `environment`.

getError() - Method in interface com.alphinat.sg5.ISmartletActionError

Obtains the error message.

getError() - Method in interface com.alphinat.sg5.ISmartletService

Gets the error message if an error occured when calling the service.

getErrorCodes() - Method in interface com.alphinat.sg5.ISmartletField

Gets the validation error codes of the field.

getErrorCodes() - Method in interface com.alphinat.sg5.ISmartletPage

Gets the page level validation error codes.

getErrorMessages() - Method in interface com.alphinat.sg5.ISmartletField

Gets the validation error messages of the field.

getErrorMessages() - Method in interface com.alphinat.sg5.ISmartletPage

Gets the page level validation error messages.

getEvent() - Method in interface com.alphinat.sg5.IServiceContext

Gets the context event.

getEventSource() - Method in interface com.alphinat.sg5.ISmartletField

Returns an array of ISmartletField which have an impact, through validation rules, dynamic values or visibility conditions on the current field.

getEventTarget() - Method in interface com.alphinat.sg5.ISmartletField

Returns an array of ISmartletField which are impacted, through validation rules, dynamic values or visibility conditions by the current field.

getFields() - Method in interface com.alphinat.sg5.ISmartletPage

Gets the page fields.

getFields() - Method in interface com.alphinat.sg5.widget.group.ISmartletGroup

Gets the top level fields of the group.

getFileExtension() - Method in interface com.alphinat.sg5.widget.upload.ISmartletUpload

Return the file extension

getFileName() - Method in interface com.alphinat.sg5.widget.upload.ISmartletUpload

Gets uploaded file name.

getFilePath() - Method in interface com.alphinat.sg5.widget.upload.ISmartletUpload

Gets uploaded file path when in disk mode for upload files.

getFileSize() - Method in interface com.alphinat.sg5.widget.upload.ISmartletUpload

Get size of uploaded file.

getFormat() - Method in interface com.alphinat.sg5.ISmartletField

Obtains the name of the format used during field validation.

getGlobalNavButtons() - Method in interface com.alphinat.sg5.ISmartlet

Gets the global navigation buttons.

getGroup(int) - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Returns the group at the specified position in the repeat.

getGroups() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Returns the groups of fields for the repeat.

getHelp() - Method in interface com.alphinat.sg5.ISmartletField

Gets the help text.

getHelp() - Method in interface com.alphinat.sg5.widget.select.ISelectOption

Gets the help text of this selection item.

getHelpId() - Method in interface com.alphinat.sg5.ISmartletField

Gets the help id used to render help link.

getHelpId() - Method in interface com.alphinat.sg5.widget.select.ISelectOption

Gets the help id used to render help link

getHint() - Method in interface com.alphinat.sg5.widget.select.ISelectOption

Gets the hint text of this selection item.

getHistory() - Method in interface com.alphinat.sg5.ISmartlet

Gets the history pages navigated by the user.

getHtmlName() - Method in interface com.alphinat.sg5.ISmartletField

Gets the html name.

getId() - Method in interface com.alphinat.sg5.ISmartlet

Obtains the unique internal identifier of the Smartlet.

getId() - Method in interface com.alphinat.sg5.ISmartletElement

Gets month of the date field, starting from 1.

getMonthString() - Method in interface com.alphinat.sg5.widget.date.ISmartletDate

Get month string of date field.

getName() - Method in interface com.alphinat.sg5.ISmartlet

Gets the Smartlet name as defined on the properties page of the Smartlet.

getName() - Method in interface com.alphinat.sg5.ISmartletField

Obtains the user-defined name of the field.

getName() - Method in interface com.alphinat.sg5.ISmartletPage

Obtains the user-defined name of the page.

getName() - Method in interface com.alphinat.sg5.ISmartletService

Gets the service name.

getName() - Method in interface com.alphinat.sg5.widget.knowledge.ISmartletKnowledgeEntry

Gets the entry name.

getNavNextButton() - Method in interface com.alphinat.sg5.ISmartletPage

Gets the next page button

getNavPreviousButton() - Method in interface com.alphinat.sg5.ISmartletPage

Gets the previous page button

getNavSummaryButton() - Method in interface com.alphinat.sg5.ISmartletPage

Gets the navigate to summary page button

getNext() - Method in interface com.alphinat.sg5.ISmartletField

Gets the next sibling field.

getPage() - Method in interface com.alphinat.sg5.ISmartletField

Gets the page that this field belongs to.

getPages() - Method in interface com.alphinat.sg5.ISmartlet

Gets the `pages` of a Smartlet as an Array.

getParent() - Method in interface com.alphinat.sg5.ISmartletEvent

Gets the nested parent event

getParent() - Method in interface com.alphinat.sg5.ISmartletField

Gets the parent field.

getParentSubSmartletField() - Method in interface com.alphinat.sg5.ISmartlet

If the current Smartlet is a subSmartlet, gets the parent subSmartlet field.

getPDFDictionarypingData(String) - Method in interface com.alphinat.sg5.ISmartlet

Gets the PDF mapping data.

getPlacement() - Method in interface com.alphinat.sg5.ISmartletField

Gets the field placement definition.

getPrefix() - Method in interface com.alphinat.sg5.ISmartletField

Gets the field prefix text.

getPrevious() - Method in interface com.alphinat.sg5.ISmartletField

Gets the previous sibling field.

getProgress() - Method in interface com.alphinat.sg5.ISmartlet

Returns the current percentage complete, from "0" to "100"

getRepeatIndex() - Method in interface com.alphinat.sg5.ISmartletField

Returns an array of integers representing repeatable group indices.

getRepeatSelectedStrings() - Method in interface com.alphinat.sg5.ISmartletField

Returns strings for selected groups in repeated field.

getRepeatSelectedValues() - Method in interface com.alphinat.sg5.ISmartletField

Returns values for selected groups in repeated field.

getRepeatStrings() - Method in interface com.alphinat.sg5.ISmartletField

Returns strings for repeated field.

getRepeatValues() - Method in interface com.alphinat.sg5.ISmartletField

Returns values for repeated field.

getRequest() - Method in interface com.alphinat.sg5.ISmartletEnvironment

Gets the Request object associated to the environment.

getResponse() - Method in interface com.alphinat.sg5.ISmartletEnvironment

Gets the Response object associated to the environment.

getResult() - Method in interface com.alphinat.sg5.ISmartletService

Gets the service call result.

getResult(String) - Method in interface com.alphinat.sg5.ISmartletService

For web services, parameter "key" is xpath (namespace ignored).

getResults(String) - Method in interface com.alphinat.sg5.ISmartletService

For web services, parameter "key" is xpath (namespace ignored).

getSelectedGroupIndexes() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Gets an integer array of the selected rows.

getSelectedGroups() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Returns the selected groups of fields for the repeat.

getSelectedLabel() - Method in interface com.alphinat.sg5.widget.select.ISmartletSelectField

Returns label of selected option.

getSelectedLabels() - Method in interface com.alphinat.sg5.widget.select.ISmartletSelectField

Returns labels of selected options.

getSelectedOption() - Method in interface com.alphinat.sg5.widget.select.ISmartletSelectField

Returns selected option.

getSelectedOptions() - Method in interface com.alphinat.sg5.widget.select.ISmartletSelectField

Returns array of selected options.

getSelectOptions() - Method in interface com.alphinat.sg5.widget.select.ISmartletSelectField

Returns the option list for the select type field.

getServices() - Method in interface com.alphinat.sg5.ISmartlet

Gets the services of the Smartlet.

getShownFields() - Method in interface com.alphinat.sg5.ISmartletPage

Gets the top level of page available fields.

getShownFields() - Method in interface com.alphinat.sg5.widget.group.ISmartletGroup

Gets the available fields directly under the group.

getShownPages() - Method in interface com.alphinat.sg5.widget.summary.ISmartletSummary

getTypeConst() - Method in interface com.alphinat.sg5.ISmartletElement

Gets the type of Smartlet element.

getTypeConst() - Method in interface com.alphinat.sg5.ISmartletField

Gets the type constant.

getTypeDetail() - Method in interface com.alphinat.sg5.ISmartletField

Gets the detailed type information of Smartlet field.

getUnSelectedGroupIndexes() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Gets an integer array of the non selected rows.

getUnSelectedGroups() - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat

Returns the non selected groups of fields for the repeat.

getUserPrincipal() - Method in interface com.alphinat.sg5.ISmartletEnvironment

Gets the principal reference containing the name of the current user.

getValue() - Method in interface com.alphinat.sg5.ISmartletField

Obtains the value of the field.

getValue() - Method in interface com.alphinat.sg5.widget.knowledge.ISmartletKnowledgeEntry

Gets the knowledge entry value.

getValue() - Method in interface com.alphinat.sg5.widget.select.ISelectOption

Gets the value of this selection item.

getValueParseScript() - Method in interface com.alphinat.sg5.ISmartletField

Gets script to parse the value

getWorkspace() - Method in interface com.alphinat.sg5.ISmartlet

Gets current workspace name

getYear() - Method in interface com.alphinat.sg5.widget.date.ISmartletDate

Gets year of the date field.

gotoPage(long) - Method in interface com.alphinat.sg5.ISmartlet

Navigate to page with given page id and add current page to history.

gotoPage(String) - Method in interface com.alphinat.sg5.ISmartlet

Navigate to page with given page name and add current page to history.

gotoPage(String, bool) - Method in interface com.alphinat.sg5.ISmartlet

Navigate to page with given page name and add current page to history.

gotoPage(String, bool, bool, bool) - Method in interface com.alphinat.sg5.ISmartlet

Navigate to page with given page name and add current page to history.

gotoPage(String, bool, bool, bool, string[]) - Method in interface com.alphinat.sg5.ISmartlet

Navigate to page with given page name.

gotoSmartlet(String, string, bool, bool) - Method in interface com.alphinat.sg5.ISmartlet

Navigate to Smartlet with given code.

GROUP - Static variable in interface com.alphinat.sg5.Constants.ElementType

The element is a group widget.

ISmartletDate - Interface in com.alphinat.sg5.widget.date

`ISmartletDate` is an interface representing a Smartlet date field.

ISmartletElement - Interface in com.alphinat.sg5

A Smartlet element can be a `ISmartlet`, `ISmartletPage`, `ISmartletField`, `ISmartletService`.

ISmartletElementVisitor - Interface in com.alphinat.sg5

Implements the Hierarchical Visitor Pattern to traverse Smartlet elements.

ISmartletEnvironment - Interface in com.alphinat.sg5

`Environment` is an interface used to obtain the Smartlet application environment.

ISmartletEvent - Interface in com.alphinat.sg5

`ISmartletEvent` is an interface representing a Smartlet event, like a click on a button.

ISmartletField - Interface in com.alphinat.sg5

`ISmartletField` is an interface representing a Smartlet field along with the operations that may be performed on that field.

ISmartletGroup - Interface in com.alphinat.sg5.widget.group

`ISmartletGroup` is an interface representing a Smartlet group.

ISmartletKnowledge - Interface in com.alphinat.sg5.widget.knowledge

`ISmartletKnowledge` is an interface representing a Smartlet knowledge widget.

ISmartletKnowledgeEntry - Interface in com.alphinat.sg5.widget.knowledge

`ISmartletKnowledgeEntry` is an interface representing a Smartlet knowledge entry.

ISmartletPage - Interface in com.alphinat.sg5

`ISmartletPage` is an interface representing a Smartlet page along with the operations that may be performed on that page.

ISmartletRepeat - Interface in com.alphinat.sg5.widget.repeat

`ISmartletRepeat` is an interface representing a Smartlet repeat widget.

ISmartletSelectField - Interface in com.alphinat.sg5.widget.select

`ISmartletField` is an interface representing a Smartlet select type field.

ISmartletService - Interface in com.alphinat.sg5

`ISmartletService` is an interface representing a Smartlet service and covers extension functions, SOAP and REST web services.

ISmartletSummary - Interface in com.alphinat.sg5.widget.summary

`ISmartletSummary` is an interface representing a summary section.

ISmartletUpload - Interface in com.alphinat.sg5.widget.upload

`ISmartletUpload` is an interface representing a Smartlet upload field.

isMultipleControls() - Method in interface com.alphinat.sg5.widget.date.ISmartletDate

Returns true if date field is shown as multiple controls.

isOptionGroup() - Method in interface com.alphinat.sg5.widget.select.ISelectOption

Returns true if the option is an option group.

isPersistent() - Method in interface com.alphinat.sg5.ISmartletField

Gets the "persist" flag.

isReadonly() - Method in interface com.alphinat.sg5.ISmartletField

Determines whether the field is read-only or not.

Sets the "persist" flag

setPlacement(String) - Method in interface com.alphinat.sg5.ISmartletField
Sets the field placement definition.

setPosition(int, int) - Method in interface com.alphinat.sg5.ISmartletField
Set field position to new row, column of current layout (row and column starts with 0).

setPrefix(String) - Method in interface com.alphinat.sg5.ISmartletField
Sets the field prefix text.

setReadonly(bool) - Method in interface com.alphinat.sg5.ISmartletField
Sets the "readonly" flag

setReadonly(bool, Collection<String>) - Method in interface com.alphinat.sg5.ISmartletField
Sets the "readonly" flag

setReadonly(bool) - Method in interface com.alphinat.sg5.widget.repeat.ISmartletRepeat
Sets the "readonly" flag for all fields in all groups of this repeat.

setRepeatStrings(String[]) - Method in interface com.alphinat.sg5.ISmartletField
Sets the repeated field with strings.

setRepeatValues(Object[]) - Method in interface com.alphinat.sg5.ISmartletField
Sets the repeated field with values.

setString(String) - Method in interface com.alphinat.sg5.ISmartletField
Sets the `string` value of the field.

setSuffix(String) - Method in interface com.alphinat.sg5.ISmartletField
Sets the field suffix text.

setTitle(String) - Method in interface com.alphinat.sg5.ISmartletPage
Specifies the page title.

setTitle(String, string) - Method in interface com.alphinat.sg5.ISmartletPage
Specifies the page title for a specific locale.

setTooltip(String) - Method in interface com.alphinat.sg5.ISmartletField
Sets the tooltip text of field.

setValidateOnSummaryOnly(bool) - Method in interface com.alphinat.sg5.ISmartletField
If sets to true, the validation will happen only on summary section.

setValidationRequire(String, string) - Method in interface com.alphinat.sg5.ISmartletField
Defines required validation.

setValidationScript(String) - Method in interface com.alphinat.sg5.ISmartletField
Defines script validation

setValue(Object) - Method in interface com.alphinat.sg5.ISmartletField
Set the field value object.

setValue(String) - Method in interface com.alphinat.sg5.widget.select.ISelectOption
Sets the value for this selection item to the specified value.

setValueParseScript(String) - Method in interface com.alphinat.sg5.ISmartletField
Defines script to parse value.

setYear(int) - Method in interface com.alphinat.sg5.widget.date.ISmartletDate
Sets year of date field.

size() - Method in interface com.alphinat.sg5.widget.select.ISelectOptionList

UPLOAD - Static variable in interface com.alphinat.sg5.Constants.ElementType

The element is an upload field.

V

validate() - Method in interface com.alphinat.sg5.ISmartletField

Revalidates the field.

validate() - Method in interface com.alphinat.sg5.ISmartletPage

Validates the page.

visit(ISmartletElement) - Method in interface com.alphinat.sg5.ISmartletElementVisitor

Visit the Smartlet element.

visitEnter(ISmartletElement) - Method in interface com.alphinat.sg5.ISmartletElementVisitor

Notifies the visitor that it is entering a new element.

visitLeave(ISmartletElement) - Method in interface com.alphinat.sg5.ISmartletElementVisitor

Notifies the visitor that the element is visited.

X

XML - Static variable in interface com.alphinat.sg5.Constants.FileType

XML file

A B C D E F G H I K L M N O P R S T U V X

Overview Package Class Use **Tree** **Deprecated** **Index** **Help**

PREV NEXT **FRAMES** **NO FRAMES** **All Classes**