



# The Grantlee Template System



Stephen Kelly



**What is Grantlee**

**How does it work?**

**How will applications use it?**



## The old way

```
QString str;  
str.append("<html><h1>" + title + "</h1>");  
str.append("<div>" + content + "</div>");  
str.append("</html>");
```



## The old way

```
QString str;  
str.append("<html><h1>" + title + "</h1>");  
str.append("<div>" + content + "</div>");  
str.append("</html>");
```

## Goals

- Separate exported data from its presentation
- Simplify theming for application developers
- Use syntax of the Django Template system



# The Grantlee Template System



**Python**

**Web framework**



**C++**

**Rich client framework**



# The Grantlee Template System



**Python**

**Web framework**

**Dynamic typing**



**C++**

**Rich client framework**

**Static typing**



# The Grantlee Template System



**Python**

**Web framework**

**Dynamic typing**

**Runtime Introspection**

**Everything is an Object**



**C++**

**Rich client framework**

**Static typing**

**Compiled binaries**

**Similar interfaces for  
similar objects**



## Template Syntax

```
<html>
{# A simple comment #}
<div>{% for entity in entities %}
  {% if entity.isBook %}
    <h1>{{ entity.title|upper }}</h1>
    {% include "booktemplate.html" %}
  {% else %}
    {% include "pagetemplate.html" %}
  {% endif %}
{% endfor %}</div>
</html>
```



## Tokens

- **Text:** “<html>”, etc
- **Comment:** “{# A simple comment #}”
- **Variable (with filter):** “{{ entity.title|upper }}”
- **Tag:** “{% for entity in entities %}”,  
“{% if entity.isBook %}”, etc



## Dynamic Typing

- **QVariant ( Any type );**
- **Multiple levels of nesting**
  - **QVariantList**
  - **QVariantHash**





## Dynamic Typing

- **QVariant ( Any type );**
- **Multiple levels of nesting**
  - **QVariantList**
  - **QVariantHash**
- **Supports custom types**
- **Cast back to strong types to process.**
  - **MyType myobject = variant.value<MyType>();**





## Introspection



`{{ map.key }}`

**dictionary**

`{{ list.2 }}`

**list**

`{{ obj.prop }}`

**getattr**



**QVariantHash**

**QVariantList**

**QObject\* property**



## Introspection

```
class KJotsEntity : public QObject
{
    Q_OBJECT
    Q_PROPERTY(QString title READ title)
    Q_PROPERTY(bool isBook READ isBook)
    Q_PROPERTY(bool isPage READ isPage)
    Q_PROPERTY(QString content READ content)
    Q_PROPERTY(QVariantList entities READ entities)

public:
    bool isBook() const;
    bool isPage() const;
    QString title() const;
    QString content() const;
    QVariantList entities() const;
};
```



## Introspection

- **Build-time code generation (Meta-object compiler)**
- **Static properties**
- **Dynamic properties**
- **Query methods, enums etc.**



## Introspection

- **Build-time code generation (Meta-object compiler)**
- **Static properties**
- **Dynamic properties**
- **Query methods, enums etc.**
- **Invokable/scriptable methods**
- **Simple type queries ( `object->className();` )**
- **Fast casting (`qobject_cast`)**



## Custom Tags and Filters



**Import modules**

**`register.tag(...)`**

**Python**



**QtPlugin**

**Well defined interface**

**C++ / EcmaScript**

**Future: Python, Ruby, ...**



## Custom Tags and Filters

```
#include <grantlee/interfaces/taglibraryinterface.h>
```

```
class MyLibrary : public TagLibraryInterface
```

```
{
```

```
    Q_OBJECT
```

```
    Q_INTERFACES( Grantlee::TagLibraryInterface )
```

```
public:
```

```
    MyLibrary();
```

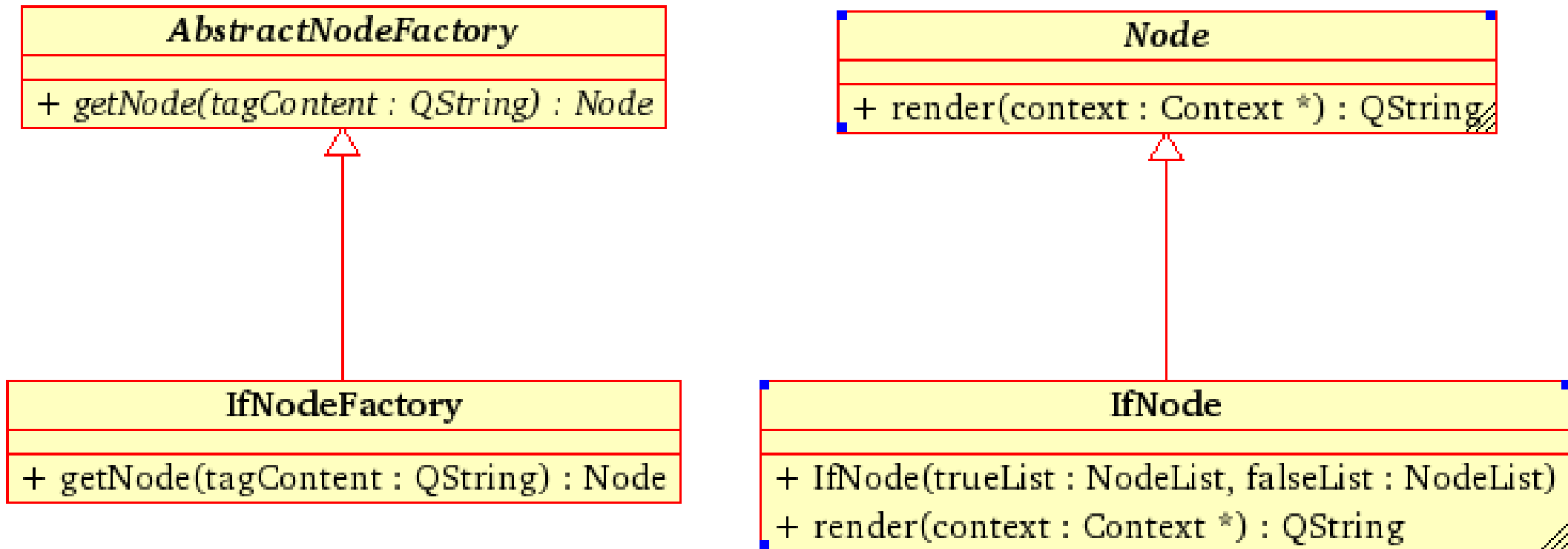
```
    QHash<QString, AbstractNodeFactory*> nodeFactories();
```

```
    QHash<QString, Filter*> filters();
```

```
};
```



## Custom Tags and Filters





# The Grantlee Template System



## Demo



# The Grantlee Template System



The screenshot shows the KJots application window titled "KJots rewrite app". The interface includes a menu bar with "Settings" and "Help", and a toolbar with "Change Theme" and "Export".

**Left Panel (Tree View):**

- Name
- Bar Book
  - + Bat Book
    - Bar Page
    - Bar2 Page
    - Bar4 Page
    - Bar3 Page
  - + Foo Book

**Right Panel (Preview):**

**Bar Book**

**Bat Book**

**Bat Page**  
Bat File content

**Bat2 Page**  
Bat2 File content

**Bar Page**  
Bar File content

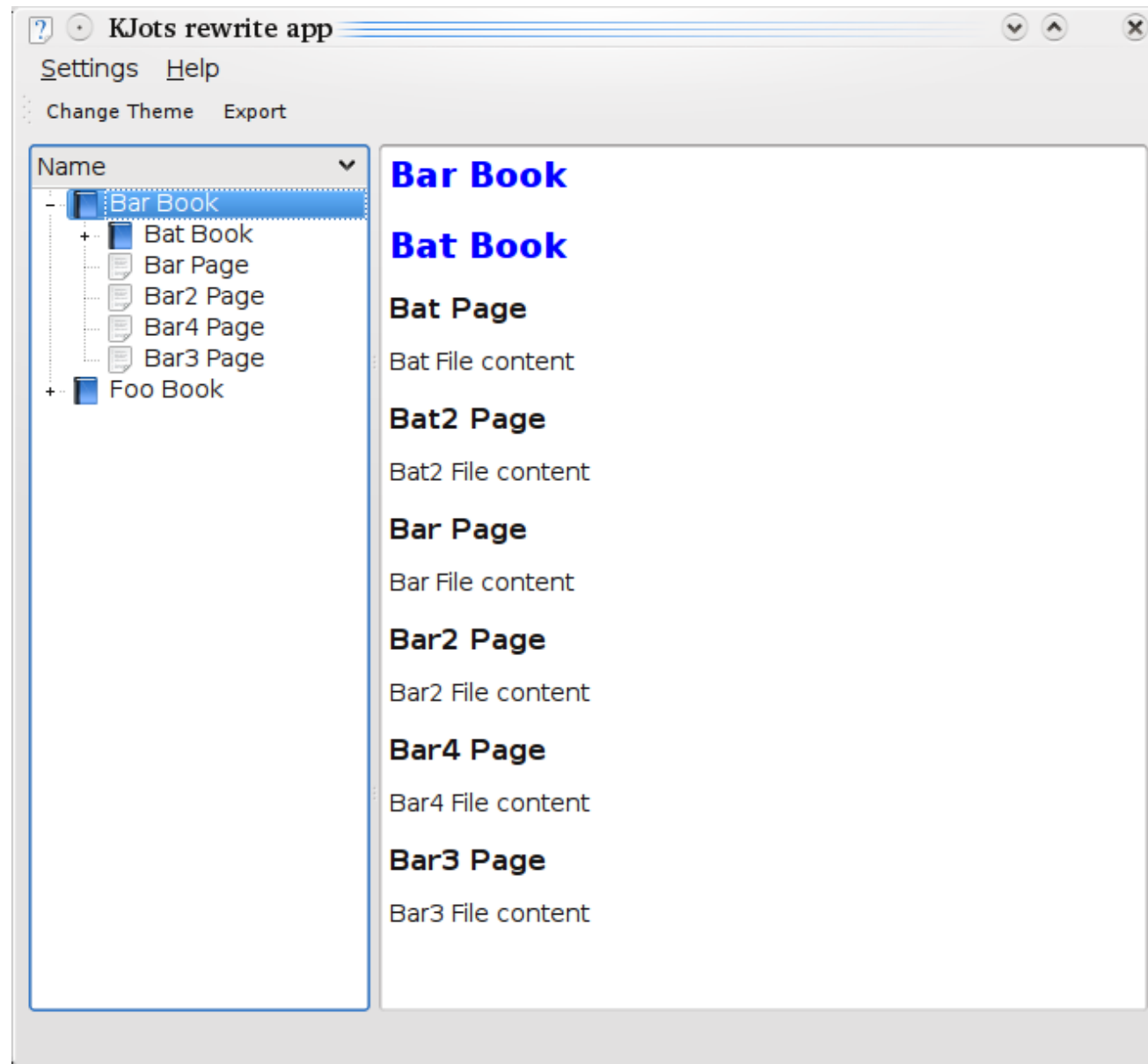
**Bar2 Page**  
Bar2 File content

**Bar4 Page**  
Bar4 File content

**Bar3 Page**  
Bar3 File content



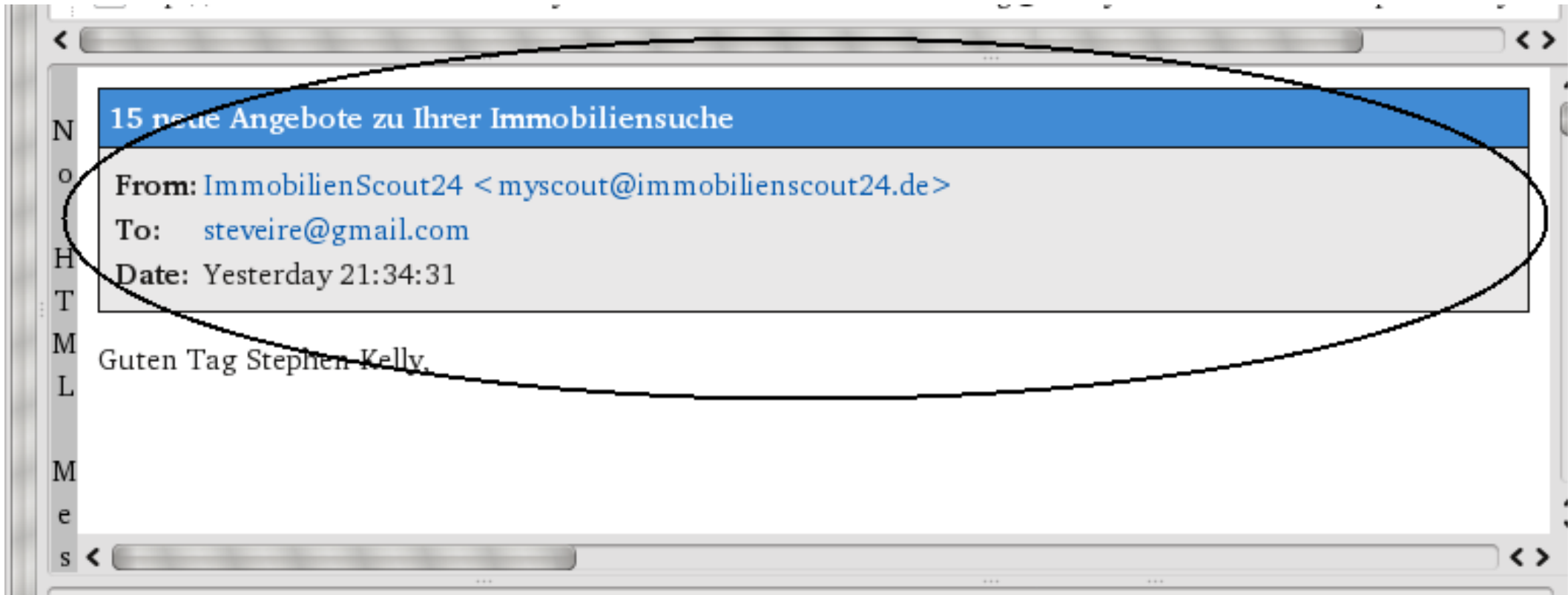
# The Grantlee Template System





## Consumers

- Kjots, KMail, KNode, Akregator, etc





## Consumers

- KJots, KMail, KNode, Akregator, etc
- Complex MIME message structures

**To: {{ recipient }}**

**Date: {% now %}**

**Subject: {{ subject }}**

**{% for header in extraHeaders %}**

**{{ header.key }}: {{header.value }}**

**{% endfor %}**

**{% for part in message %}**

**{% include part.typeTemplate %}**

**{% endfor %}**



## Consumers

- Kjets, KMail, KNode, Akregator, etc
- Complex MIME message structures
- Akonadi mail-merge agent

Dear `{{ customer.contactName }}`,

We are out of stock of `{{ product }}`, so your orders on  
`{% for order in customer.affectedOrders %}`

`* {{ order.date }}`

`{% endfor %}`

Can not be filled.



## Summary

- **Successful port of corelib and default tags**
- **Covered by LGPLv3**
- **350 unit tests ported**
- **Performance tests with QBENCHMARK**
- **Public release 'soon'**



## Questions & Answers