Indigo Platform Documentation

Release 1.0

Code for South Africa

May 18, 2018

Contents

Indigo is AfricanLII's document management system for managing, capturing and publishing legislation in the Akoma Ntoso format, built by OpenUp with a grant from Indigo Trust.

You can try it out at indigo-sandbox.openup.org.za. Login with the username and password *guest@example.com*.

Visit our GitHub repository to find out how you can contribute to the project.

The Indigo platform is a Django python web application with three components:

- a web application for managing and editing documents
- an **application REST API** upon which the web application runs
- a **public read-only REST API** for vending published legislative documents in HTML and XML

The main REST API supports the creation, editing and management of a library of legislative documents. The public read-only API provides access to published documents in XML, HTML or other formats.

CHAPTER 1

Contents

1.1 Installing and Running Indigo

This guide is for developers who want to install and run their own instance of the Indigo platform.

Indigo is a Django web application that is easily deployed on Heroku or Dokku. Dokku emulates a Heroku-like environment on your own servers (or cloud).

Note: We recommend using Dokku for production because some Indigo functionality – such as parsing new documents – can take longer than the 30 seconds Heroku allows for web API calls. However, Heroku is great for quickly trying Indigo out.

1.1.1 Requirements

Indigo requires:

- Python 2.7.8
- Postgresql 9.3+
- Ruby 2.1.6+ for Slaw
- Java 1.8 for Apache Tika
- An AWS S3 account and bucket for storing attachments

Optional but useful:

- A New Relic monitoring account
- An SMTP server to send email through, or a service like Mandrill

Using Heroku/Dokku means that we're using well-document and widely understood mechanisms for specifying dependencies and configuration across multiple languages. It takes care of all this for you and we highly recommend using them.

1.1.2 Installation

Installation on Heroku and Dokku are similar and only really differ in the commands that are run. We describe using Heroku below.

1. Create a new AWS S3 account and bucket for storing attachments. You'll need the AWS Access Key Id and AWS Secret Access Key later.
2. Clone the repo:

```
$ git clone https://github.com/Code4SA/indigo
$ cd indigo
```

3. Create a new Heroku application:

```
$ heroku apps:create indigo
```

4. Create a Postgres database:

```
$ heroku addons:create heroku-postgresql
```

5. Set config options:

```
$ heroku config:set indigo \
    DISABLE_COLLECTSTATIC=1 \
    DJANGO_DEBUG=false \
    DJANGO_SECRET_KEY=some random characters \
    AWS_ACCESS_KEY_ID=aws access key \
    AWS_SECRET_ACCESS_KEY=aws secret access key \
    AWS_S3_BUCKET=your-bucket-name
```

6. If you're using New Relic, you'll need to set those config options:

```
$ heroku config:set indigo \
    NEW_RELIC_APP_NAME=Indigo \
    NEW_RELIC_LICENSE_KEY=7e4b428f9f46d4b3a943a3577e636337f35e5ec4
```

7. Deploy:

```
$ git push heroku
```

8. Setup the Indigo database:

```
$ heroku run python manage.py migrate
```

9. Create the admin user:

```
$ heroku run python manage.py createsuperuser
```

10. Visit your new Indigo app in your browser!
11. You can configure new users and other things at */admin*.
12. You'll need to set some *Permissions* for users.

TODO:

- document configuring country details

1.1.3 Configuration

Config options are mostly passed to Indigo as environment variables. These are the options you can set:

- `AWS_ACCESS_KEY_ID`

  Required for production. The AWS access key ID for the account with write-access to the S3 bucket used for storing attachments.

- `AWS_SECRET_ACCESS_KEY`

  Required for production. The AWS secret access key for the account with write-access to the S3 bucket used for storing attachments.

- `AWS_S3_BUCKET`

  Required for production. The name of the S3 bucket for storing attachments.

- `AWS_S3_HOST`

  The regional S3 endpoint to use. Optional. Default: `s3-eu-west-1.amazonaws.com`

- `DATABASE_URL`

  Required. The URL of the database to use

- `DJANGO_DEBUG`

  The Django `DEBUG` setting. Everything other than `true` means False. This should always be `false` in production. Default: `true`

- `DJANGO_DEFAULT_FROM_EMAIL`

  The Django `DEFAULT_FROM_EMAIL` setting: who do emails come from? Uses `SUPPORT_EMAIL` by default.

- `DJANGO_EMAIL_HOST`

  The Django `EMAIL_HOST` setting. The SMTP host through which to send user emails such as password resets.

- `DJANGO_EMAIL_HOST_PASSWORD`

  The Django `EMAIL_HOST_PASSWORD` setting. The SMTP password.

- `DJANGO_EMAIL_HOST_PORT`

  The Django `EMAIL_HOST_PORT` setting. The SMTP port (default: 25).

- `DJANGO_EMAIL_HOST_USER`

  The Django `EMAIL_HOST_USER` setting. The SMTP username.

- `DJANGO_SECRET_KEY`

  Required if DJANGO_DEBUG is not true. The Django `SECRET_KEY` setting. In production you should use a random (and secret) string.

- `GOOGLE_ANALYTICS_ID`

  Google Analytics ID for website tracking. Only used when `DEBUG` is False.

- `NEW_RELIC_APP_NAME`

  The New Relic App Name, if you're using New Relic.

- `NEW_RELIC_LICENSE_KEY`

  The New Relic license key, if you're using New Relic.

- `SUPPORT_EMAIL`

  Required Email address users can email for help.

1.2 Using the Indigo Platform

Welcome to the guide to using the Indigo platform to manage and maintain a collection of legislation documents. This guide is for law librarians and others who are familiar with writing legislation documents and want to know how to use the Indigo Platform.

Note: Throughout this guide we refer to **documents**. These are generally legislative Acts but may also be related legislative documents such as regulations or by-laws.

Contents:

1.2.1 Managing Your Account

Logging In

You need to log into your Indigo account to import and edit documents. You may be able to view documents without logging in.

To log in, click the **Log in** button in the top-right corner, enter your email address and password, and click the **Log in** button.

Creating an account

New accounts must be created manually. Please email the address in the login window to get an account.

Note: Administrators: visit `/admin/auth/user/` on your indigo instance to create a new account. Always use an email address for the username.

Resetting your password

You can reset your password if you've forgotten it.

1. Click the **Log in** button in the top-right corner.
2. Click **I forgot my password**.
3. Fill in the email address you use for your account.
4. Click **Reset password**.
5. Indigo will send you an email with a link to reset your password. Click the link in the email and follow the prompts to change your password.
6. Go back to the login page, click **Log in**, enter your email address and your new password, and click **Log in**.

Changing your name or email address

You can change your profile details after you've logged in.

1. Click your button with your name or email in the top-right corner.
2. Click the **Profile** menu item.
3. Update your details.
4. Click **Save changes**.

Changing your password

You can change your password after you've logged in.

1. Click your button with your name or email in the top-right corner.
2. Click the **Profile** menu item.
3. Click **Change your password**.
4. Enter your new password twice to confirm it.
5. Click **Change password**.

Logging out

You don't need to log out.

If you need to log out,

1. Click your button with your name or email in the top-right corner.
2. Click **Log out**.

1.2.2 Indigo Principles

The Indigo platform works a bit differently to other consolidation platforms you might be familiar with. Understanding these differences will help make explain why the Indigo platform doesn't look like a text editor such as Microsoft Word.

Structure, Content and Presentation

A key goal of the Indigo platform is to allow content to be published for a wide variety of media and devices, including print, desktop web browsers, mobile phones and others that we don't yet know of. It's important that a document can be formatted appropriately for each medium. For example, a printed document will include the Table of Contents at the start of the document, while a web browser and a mobile phone will display the Table of Contents differently to help users navigate the document effectively.

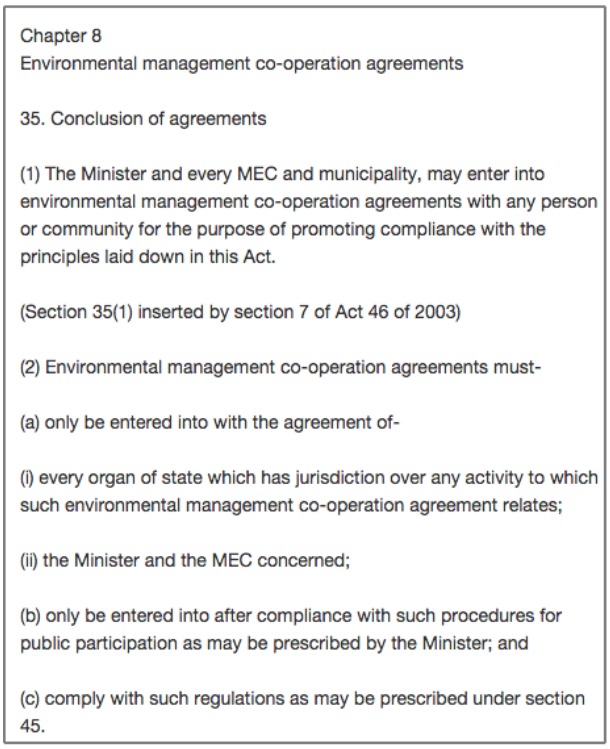
To achieve this, the platform must capture the **metadata**, **content** and **structure** of the document separately from its **presentation**.

- **Metadata** describes the document as a piece of work and includes information such as its title, publication date, language and country. This data allows users to classify, browse and search for documents and is also used when presenting the document.
- **Structure** is the heirarchy of the document, including chapter, part and section information. Capturing the structure makes it possible to provide intuitive user experiences, such as an interface for browsing through a document by section or linking between different sections of a document or between documents.

- **Content** is the actual words of the document, including headings, tables and paragraph numbers, and forms the core of the document.
- **Presentation** is how the document is physically presented to the user and includes typography, layout, colour and formatting.

Note: Notice that presentation is separate from structure and comes *last* in the logical pipeline of managing a document. The same metadata, content and structure can be presented in many different ways.

Here is an example piece of legislation that illustrates the difference between content, structure and presentation.



Chapter 8
Environmental management co-operation agreements

35. Conclusion of agreements

(1) The Minister and every MEC and municipality, may enter into environmental management co-operation agreements with any person or community for the purpose of promoting compliance with the principles laid down in this Act.

(Section 35(1) inserted by section 7 of Act 46 of 2003)

(2) Environmental management co-operation agreements must-

(a) only be entered into with the agreement of-

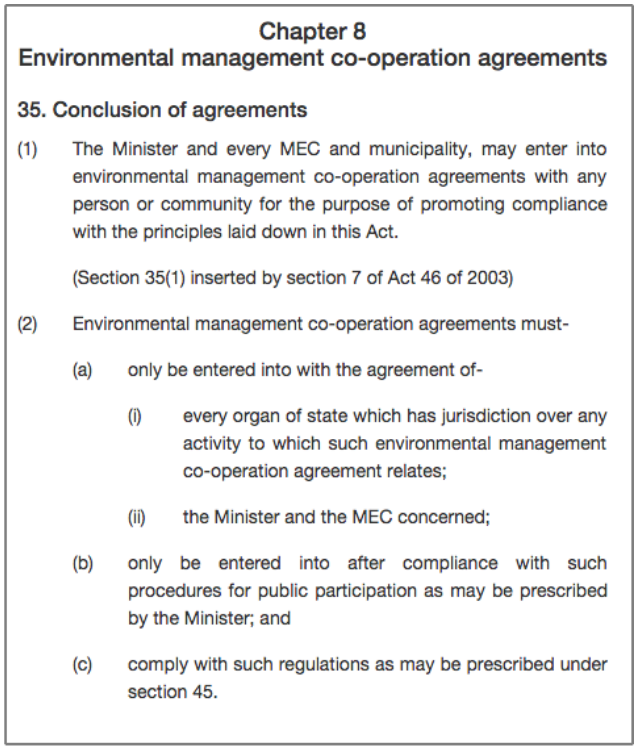
(i) every organ of state which has jurisdiction over any activity to which such environmental management co-operation agreement relates;

(ii) the Minister and the MEC concerned;

(b) only be entered into after compliance with such procedures for public participation as may be prescribed by the Minister; and

(c) comply with such regulations as may be prescribed under section 45.

Content without structure
with simple presentation



Chapter 8
Environmental management co-operation agreements

35. Conclusion of agreements

(1) The Minister and every MEC and municipality, may enter into environmental management co-operation agreements with any person or community for the purpose of promoting compliance with the principles laid down in this Act.

(Section 35(1) inserted by section 7 of Act 46 of 2003)

(2) Environmental management co-operation agreements must-

(a) only be entered into with the agreement of-

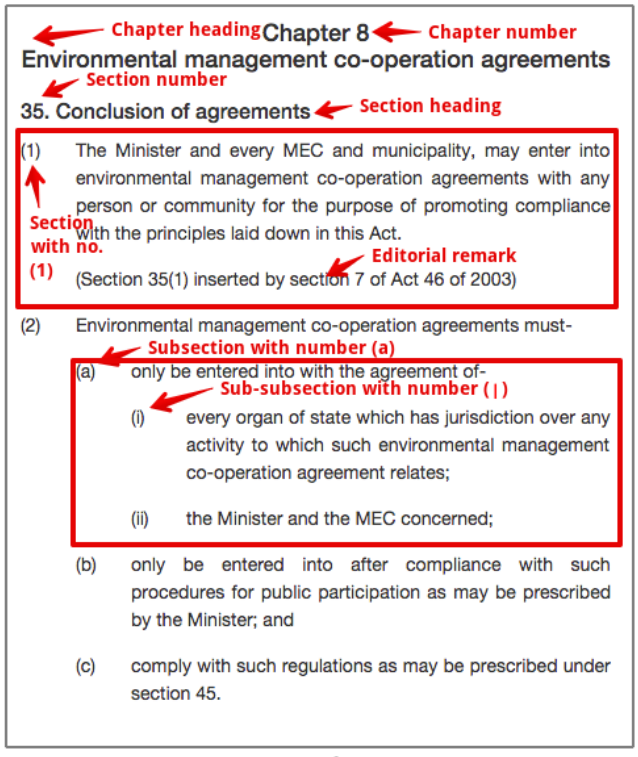
(i) every organ of state which has jurisdiction over any activity to which such environmental management co-operation agreement relates;

(ii) the Minister and the MEC concerned;

(b) only be entered into after compliance with such procedures for public participation as may be prescribed by the Minister; and

(c) comply with such regulations as may be prescribed under section 45.

Content with presentation
based on structure



Structure of content

The Indigo platform focuses on capturing the metadata, structure and content of documents. It provides some support for basic presentation of a document–primarily HTML–and allows others to build on the captured structure to provide

new presentation layers. This means the hard work done to capture a document in Indigo will keep paying off over time as the support for the underlying data format grows.

Note: It's more important to capture the **content** and **structure** of a document than exactly match the presentation of the original.

A good example of this separation of content, structure and presentation is the guide you're reading right now. You can read this guide in at least four different forms, each with its own style of presentation:

- As a website
- As a PDF
- As a single downloadable webpage
- As an Epub e-book

This is all possible because the guide is written using reStructured Text, a simple text format for writing documentation that focuses on content and structure and ignores presentation. Here's what the reStructured Text source for this section looks like::

```
Structure, Content and Presentation
-----------------------------------

A key goal of the Indigo platform is to allow content to be published for a wide␣
↪variety of media and devices, including print, desktop web browsers, mobile phones␣
↪and others that we don't yet know of. It's important that a document can be␣
↪formatted appropriately for each medium. For example, a printed document will␣
↪include the Table of Contents at the start of the document, while a web browser and␣
↪a mobile phone will display the Table of Contents differently to help users␣
↪navigate the document effectively.

To achieve this, the platform must capture the **metadata**, **content** and␣
↪**structure** of the document separately from its **presentation**.

- **Metadata** describes the document as a piece of work and includes information␣
↪such as its title, publication date, language and country. This data allows users␣
↪to classify, browse and search for documents and is also used when presenting the␣
↪document.
- **Structure** is the heirarchy of the document, including chapter, part and section␣
↪information. Capturing the structure makes it possible to provide intuitive user␣
↪experiences, such as an interface for browsing through a document by section or␣
↪linking between different sections of a document or between documents.
- **Content** is the actual words of the document, including headings, tables and␣
↪paragraph numbers, and forms the core of the document.
- **Presentation** is how the document is physically presented to the user and␣
↪includes typography, layout, colour and formatting.

.. note::

    Notice that presentation is separate from structure and comes *last* in the␣
↪logical pipeline of managing a document. The same metadata, content and structure␣
↪can be presented in many different ways.

Here is an example piece of legislation that illustrates the difference between␣
↪content, structure and presentation.

.. image:: content-vs-structure.png
```

A **compiler** converts this plain text into the different formats listed above, presenting it in the best way for each format. As the author, all I need to do is ensure the content and the structure are well expressed in the reStructured Text format, and the compiler does all the hard work to make it look good. I don't need to worry about styling the document at all.

See also:

You can see the full reStructuredText source for this guide at https://github.com/Code4SA/indigo/tree/master/docs.

Akoma Ntoso

Under the hood, Indigo uses the Akoma Ntoso standard for legal documents. This is an XML standard that allows us to capture the content and–most importantly–the structure of the document. Akoma Ntoso supports a wide range of documents (acts, bills, debates, gazettes, etc.) but we only use the **act** document type. Documents such as by-laws, statutory instruments and government notices also fall under this type.

See also:

See http://www.akomantoso.org/ for more background on Akoma Ntoso.

Akoma Ntoso is designed to support the many varying structures of legislative documents used throughout the world. As a result, the format is rich and expressive but quite complicated. Indigo works with only a subset of what is allowed by Akoma Ntoso and does its best to hide this complexity from the user.

It can be useful to understand what Akoma Ntoso looks like. Here is the XML that corresponds with the content and structure from above:

```
<chapter xmlns="http://www.akomantoso.org/2.0" id="chapter-8">
  <num>8</num>
  <heading>Environmental management co-operation agreements</heading>
  <section id="section-35">
    <num>35.</num>
    <heading>Conclusion of agreements</heading>
    <subsection id="section-35.1">
      <num>(1)</num>
      <content>
        <p>The Minister and every MEC and municipality, may enter into environmental man
      </content>
    </subsection>
    <subsection id="section-35.subsection-1">
      <content>
        <p>(Section 35(1) inserted by section 7 of Act 46 of 2003)</p>
      </content>
    </subsection>
    <subsection id="section-35.2">
      <num>(2)</num>
      <content>
        <blockList id="section-35.2.list2">
          <listIntroduction> Environmental management co-operation agreements must- </li
          <item id="section-35.2.list2.a">
            <num>(a)</num>
            <blockList id="section-35.2.list2.a.list0">
              <listIntroduction>only be entered into with the agreement of-</listIntrodu
              <item id="section-35.2.list2.a.list0.i">
                <num>(i)</num>
                <p>every organ of state which has jurisdiction over any activity to whic
              </item>
              <item id="section-35.2.list2.a.list0.ii">
                <num>(ii)</num>
                <p>the Minister and the MEC concerned;</p>
              </item>
            </blockList>
          </item>
          <item id="section-35.2.list2.b">
            <num>(b)</num>
            <p>only be entered into after compliance with such procedures for public par
          </item>
```

As you can see, the Akoma Ntoso XML for a section is complicated! If you understand the format you can edit the XML directly, otherwise we recommend you use the editor.

What Indigo Does for You

Indigo automates some parts of managing legislation. It can do this because it understands the structure of the document.

1. Indigo generates the cover page for a document based on the metadata you supply. You must not include a cover page in the content of the document.
2. Indigo generates a full Table of Contents with Parts, Chapters, Sections and Schedules. You must not add a Table of Contents to you document; it's important that you don't so that Indigo doesn't get confused.

3. Indigo manages typefaces and font sizes for you. This is a presentation issue and is therefore dependent on what device or media the document is being targeted at. Indigo uses the structural information you provide to decide how the document should look on different devices.
4. Indigo handles indentation for you, based on the document structure. It knows when a subsection is a child of a section and will sort out the indentation appropriately, you don't need to worry. In fact, Indigo completely ignores tabs and spaces when it imports documents.

Next, we'll take you through how to use the Indigo platform to manage and capture legislation.

1.2.3 Works and the Library

A work is an Act, a by-law, a regulation, and so on. In Indigo, a work doesn't have any content – it's just a description of the basic details of the act (or by-law, etc.). A work can be associated with many documents, each of which is an *expression* of the work.

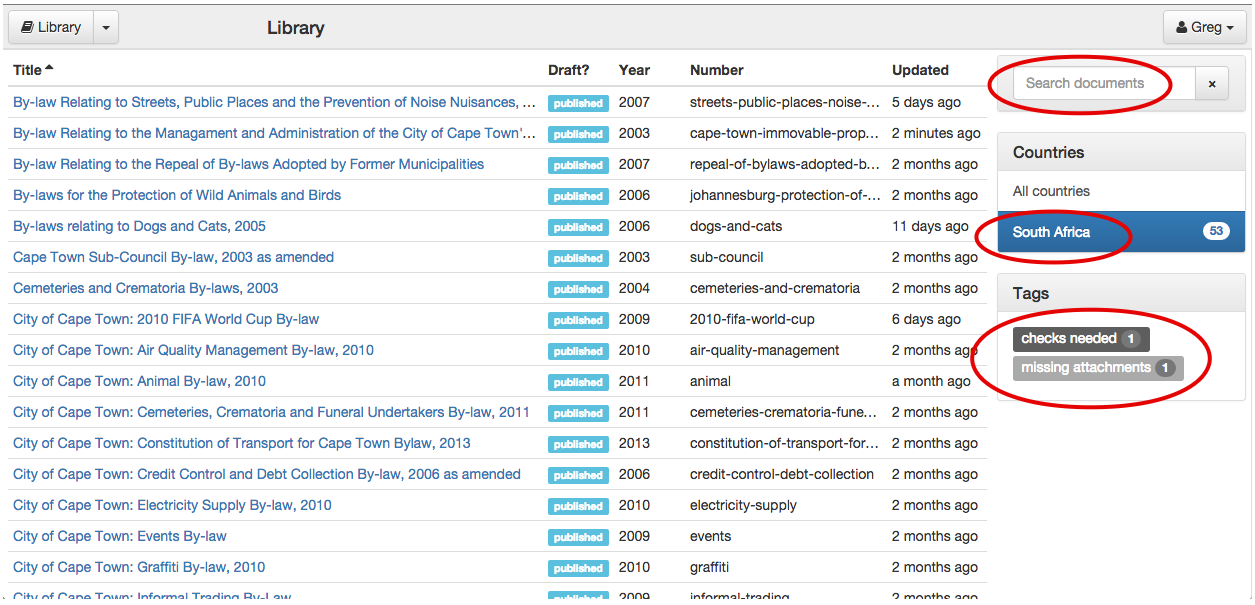
An Indigo work is uniquely identified by its FRBR URI. Documents are linked to a work through a common FRBR URI.

Documents inherit a number of fields from a work, such as the FRBR URI, publication date, repeal status, etc. These details are managed at the work level, not at the document level.

You must create a work before you can import a new document.

Library

The library lists all the works and documents in the system. You can filter them by country and by tag and search for them by title, year and number.



Searching

Search for a document by typing in the search box. This will limit the documents to only those that match your search. Searches ignore case and match against title, year and number. Clear the search by clicking the **x** button.

Filtering by Country

To show only documents for a particular country, choose the name of the country from the list.

Filtering by Tag

Tags are a powerful way to group and manage documents however you need to. You can add as many tags as you like to a document.

Change a document's tags by clicking on the document name and changing them in the **Basic details** section of the Properties page.

You can filter the documents to show only those with one or more tags. To do so, click the tag in the list of tags on the right. The number next to the tag is the number of documents that have that tag. Tags you are filtering by are highlighted in blue.

If you choose to filter by multiple tags, only documents with all of the chosen tags will be shown.

Editing Works

To edit a work, click through to the work from the library or a linked document.

To create a new work, use the **New Work** option in the library menu.

Work Details

Short title is the generally used title of the work. Most pieces of legislation declare what the short title will be.

Country the country that this legislation is applicable to.

Locality (optional) the area within the country that this legislation applies to. This is not applicable for national legislation and can be left blank. If used, this should be a widely accepted code for the region. For example, in South Africa the list of municipality codes is available on Wikipedia.

Document subtype (optional) is the subtype of the work. Choose **(none)** for general Acts.

Year is the year of the work, generally the year it was first introduced in Parliament.

Number is the number of the work within the year. Most Acts are assigned a sequential number within the year of introduction. If you don't have a number available (eg. for by-laws) use a reasonable short form of the work's name or, as a last resort, use `nn` for *not numbered*. Use `cap123` for Chapter 123 in a Cap-based numbering system.

FRBR Work URI is the URI used to identify this work and is calculated from the other metadata. It uniquely identifies this work. You cannot edit this value.

Note: Administrators can add new countries and subtypes through the Admin interface. Click on your name in the top-right corner and choose **Site Settings**.

Promulgation

Publication date (optional) is the date on which this work was officially published, in the format `YYYY-MM-DD`.

Publication name (optional) is the name of the publication in which it was published, such as *National Gazette*.

Note: Administrators can set the publication names which are available in the drop-down list. Click on your name in the top-right corner and choose **Site Settings**, then edit a country in the Indigo Editor section.

Publication number (optional) is the number of the publication in which it was published, such as a gazette number.

Assent date (optional) is the date on which the President or other authority assented to the work, in the format `YYYY-MM-DD`.

Commencement date (optional) is the date on which the bulk of the work comes into force, in the format `YYYY-MM-DD`. If parts of the work come into force on different dates, place an editorial comment inline in the document indicating this.

Repeal

Repealing work The work which repealed this work, if any.

Repeal date (optional) is the date on which the work was repealed, in the format `YYYY-MM-DD`.

1.2.4 Managing Documents

Importing a new document

You can create a new document by importing an existing file, such as a PDF or a Word document. Indigo uses Apache Tika to import from a wide range of document types, including:

- MS Word (.doc and .docx)
- Rich Text Format (.rtf)
- PDF
- Plain text (.txt)

Simple documents such as Word (.doc) and RTF produce the best results.

Note: Follow these tips for getting the best results when importing documents:

- Prefer RTF or Word (.doc and .docx) documents; use PDFs only as a last resort
- Remove the Table of Contents at the start of the document
- Convert images to text

To import a document:

1. Click the arrow next to the **Library** button and choose **Import a document**.
2. Drag and drop the file to import into the box, or click the button to choose a file to upload.
3. Wait for the document to be imported. This make take up to 10 minutes, especially for large documents.

Once you have imported a document you will need to proof it to ensure that the various components have been correctly captured. Indigo doesn't always get everything right. Look for these errors after importing:

- Check that parts and chapters have been identified correctly.
- Check that numbered sections have been identified correctly.

- Check that numbered lists aren't broken in the wrong places.
- Check that schedules have been matched correctly.

See also:

See the section on *Editing Documents* for more details.

Indigo will do its best to extract text information from the imported document. You will still need to fill in all the metadata such as the document title, year of publication, etc.

Deleting a document

You can delete a document by going to the document Properties page and scrolling down to the **Danger Zone** section and clicking the **Delete this document** button.

Note: You cannot delete a published document. Mark it as a draft first.

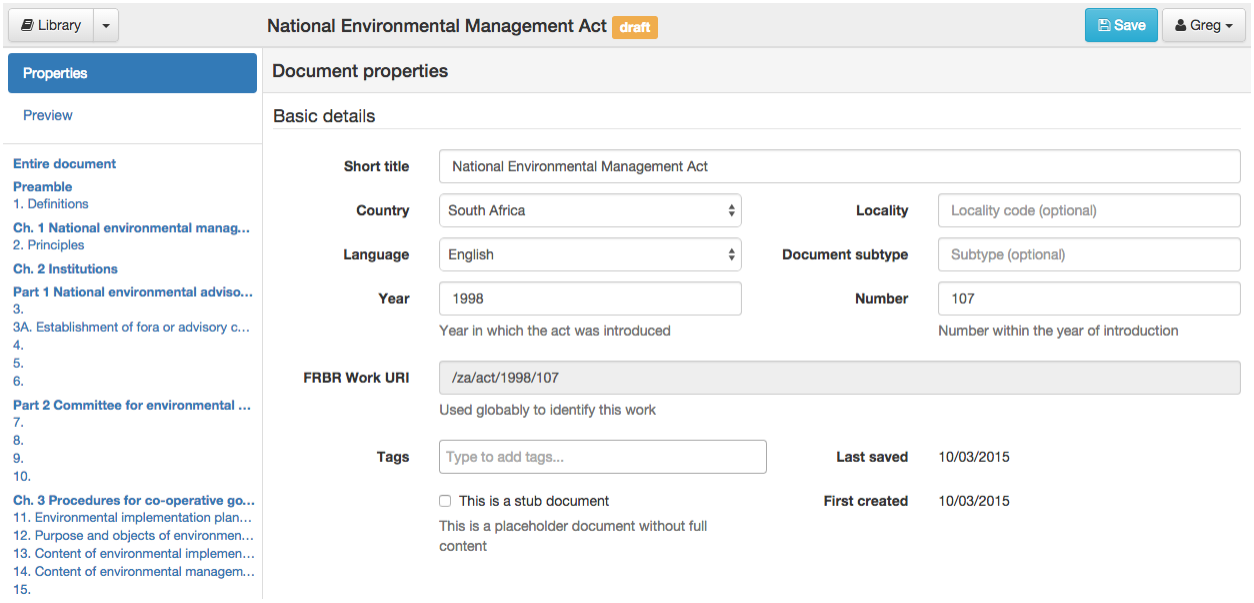
Note: Only users with the **Can delete document** permission can delete a document. An Administrator can change this for you in the Admin area.

Note: If you delete a document by accident an administrator can undelete it for you.

Administrators: visit `/admin/indigo_api/document/` and click on the document to recover, scroll down to find the **Deleted** checkbox, uncheck it and click **Save**.

Editing Metadata

The metadata is important for describing the document and making it available through the API.



Basic Details

Work The details of the work this document is linked to. Change the work by clicking the button.

Short title is the generally used title of the document. Most pieces of legislation declare what the short title will be.

Language is the language of the document.

Tags is a free-form collection of tags for this document. Use tags to manage your documents during editing and even after publication. To add a new tag, click in the box, type a new tag and press enter or comma. You can add as many tabs as you like. To delete a tag, either backspace or click the **x** next to the tag's name.

Stub document indicates that the document doesn't have all its content yet. This is useful when other documents reference this one but no source is available or the source has not been fully checked.

Note: Administrators can add new countries, languages and document subtypes through the Admin interface. Click on your name in the top-right corner and choose **Site Settings**.

Draft and Publishing

Draft controls whether the document is available publically. While you are editing the document, this should be **checked**. Outside users cannot see draft documents. Once a document is ready to be used by outside users, uncheck this box to indicate it is published.

Note: You cannot delete a published document. Mark it as a draft, first.

Amendments

The Amendments section records amendments that have been applied to reach **this version of the document**. If you are not editing the latest version of the document this must only include those amendments that have been applied to reach this point.

To record an amendment, you need the following information about the **amending document** (the document that caused the amendments to happen):

- the title
- date of publication (date at which the amendments took place)
- the FRBR URI of the document

If the amending document is already in the library, you can choose it from the list and have all these details filled in automatically.

To create a newly amended version of a document, edit the version just before the new amendments need to be applied and click **Clone Document** to create a copy, and then edit that copy.

Attachments

The Attachments section lets you attach files to your document. This is generally used to link a source PDF, Word or other file with your document, but you can upload any file you like. When creating a new document by importing a file, the file is automatically added as an attachment.

To upload a new file, click on **Attachments** and then click the **Choose Files** button.

You can change the name of an attachment by clicking the pencil (edit) icon.

Defined Terms Analysis

Indigo can find defined terms in a document and associate occurrences of a term with its definition. It does this by looking for a section called `Definitions` or `Interpretation` and looking for sentences that look like definitions. It then looks through the document to find uses of any defined terms it has found.

To find and link terms, click **Analysis** and then **Find and link defined terms**.

When viewing a document, Indigo marks the definition of a defined term **in bold**.

Important: Defined terms are lost when a section is edited. It's best to find and link defined terms just before publishing a final document, or after doing a batch of updates.

1.2.5 Editing Documents

If you're capturing a new document we recommend first capturing it using a standard word processor like Microsoft Word. Once you have the bulk of the document, import it into Indigo.

Indigo looks for structure in a document by looking for keywords such as *chapter* and *part*. It can identify different aspects of a document:

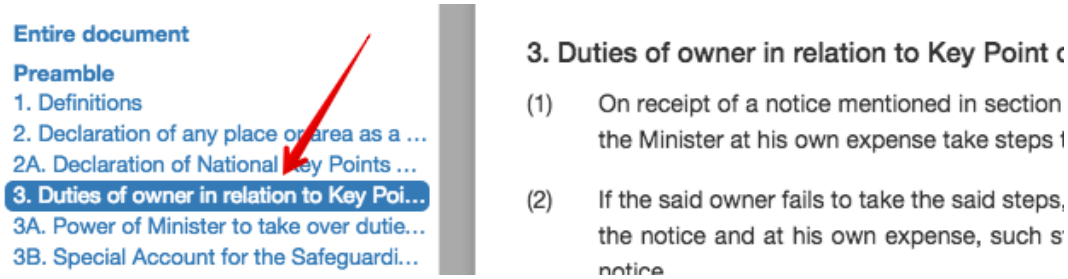
- Preface
- Preamble
- Parts and chapters
- Sections, subsections and numbered lists
- Schedules

Note: Indigo ignores presentation details such as font size, bold and italicised text, and indentation because those elements are used inconsistently by different authors. Indigo will apply new presentation rules based on the structure of the document.

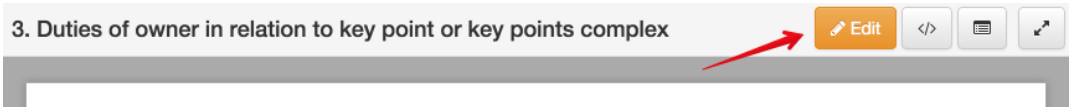
Making and Saving Changes

The easiest way to make an edit is to use the Table of Contents on the left part of the document page to find the section in which to make the change.

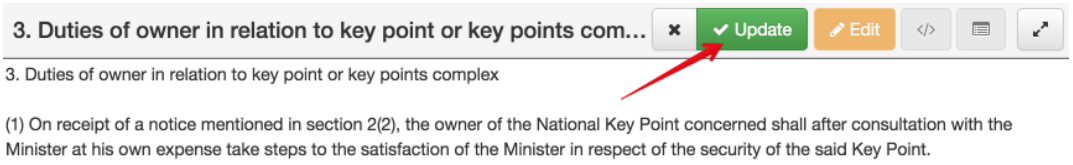
1. Click on the heading of the part, chapter or section you wish to edit. Choose the smallest element that contains what you wish to change.



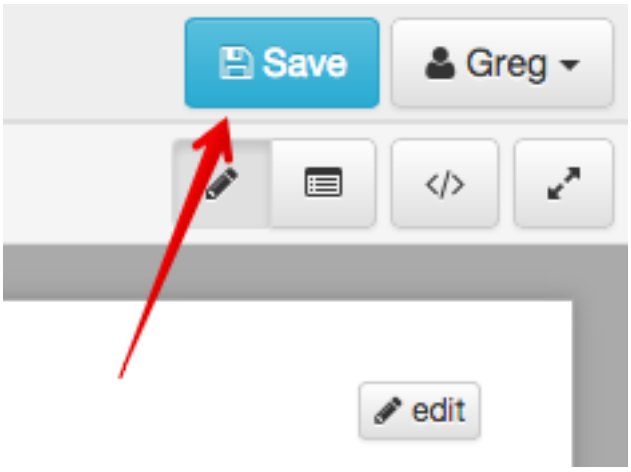
2. Click the **Edit** button in the top right corner of the document.



3. Indigo will show the simple editor. Notice that the content of the editor is the textual content of the section you're editing, without any formatting and with very simple layout.
4. Make the changes you require.
5. Click the **green tick** at the top-right corner where you clicked the **Edit** button.



6. Indigo will process your change and replace the editor with the new content.
   - If you've made an edit Indigo cannot understand, clicking the **green tick** will show an error. Correct your edit and try again.
   - To abandon your changes, click the **X** icon near the green tick.
8. Click the blue **Save** button to save your changes to the server.



Note: Bear these tips in mind when editing:

- Indigo can take a long time to process large sections. Choose the smallest containing element when editing.
- Use the existing content as a guide for how to format new content.

Editing Structure

Indigo cares about the structure of a document, not font sizes or layout. This means you need to follow a few simple conventions and Indigo will do all the hard work for you.

Here is an example of the simple formatting used by Indigo:

```
Chapter 8 - Environmental management co-operation agreements

35. Conclusion of agreements

(1) The Minister and every MEC and municipality, may enter into environmental␣
↪management co-operation agreements with any person or community for the purpose of␣
↪promoting compliance with the principles laid down in this Act.

[[Section 35(1) inserted by section 7 of Act 46 of 2003]]

(2) Environmental management co-operation agreements must-

(a) only be entered into with the agreement of-

(i) every organ of state which has jurisdiction over any activity to which such␣
↪environmental management co-operation agreement relates;

(ii) the Minister and the MEC concerned;

(b) only be entered into after compliance with such procedures for public␣
↪participation as may be prescribed by the Minister; and

(c) comply with such regulations as may be prescribed under section 45.
```

Indigo understands how to convert this into the XML that represents a chapter, section 35, subsections etc.

Notice that under subsection 1(a) above there is a sublist with items (i) and (ii). We don't bother trying to indicate that it is a sublist, Indigo will work that out based on the numbering.

You can think of this as focusing on the **content** of the document and using very simple **presentation** rules guided by a rough understanding of the **structure**. Compare this with an editor like Word which focuses heavily on the **presentation** of the content.

Chapters, Parts, Sections, etc.

Follow these guidelines to tell Indigo about the structure of your document:

- Start the preface like this:

```
PREFACE
```

- Start the preamble like this:

```
PREAMBLE
```

- Start a chapter like this:

```
Chapter 2 - Interpretation
```

- Start a part like this:

```
Part 1 - Applications
```

- Start a section like this:

```
1. Definitions
```

- Numbered subsections must have a number in parentheses at the start of the line:

```
(1) The content of section 1.

(2) The content of section 2.
```

- Subsections or statements without numbers can be written as-is:

```
A statement without a number.
```

- Numbered sublists must have a number in parentheses at the start of the line:

```
(a) sublist item a

(b) sublist item b
```

- Start a Schedule like the example below. The number `1`, the `Title` and `Heading` parts are optional, you can leave them out if necessary:

```
Schedule 1 - Title
Heading

or

Schedule - First Schedule
Subtitle
```

Tables

Often a piece of legislation will include tables, for example in Schedules.

The easiest way to edit these is to click the **Edit Table** button at the top right corner of the table.

- Simply type your changes into the table and click **Update** when you're done.
- Use the buttons in the toolbar to add and remove columns and rows, and to set cells as heading cells.

- To **merge** cells, use the mouse to select the cells and click **Merge cells**.
- To **split merged cells**, select the cells and click the **Merge cells** button again.

Editing Tables in the Simple Editor

You can also edit tables in the simple editor. Indigo uses the same text format for tables that Wikipedia uses.

See also:

Be sure to read Wikipedia's tutorial for writing tables.

Use this table generator website to easily start building a new table for a document.

Don't use `class="wikitable"` even though they recommend it.

This code:

```
{|
|-
! header 1
! header 2
! header 3
|-
| row 1, cell 1
| row 1, cell 2
| row 1, cell 3
|-
| row 2, cell 1
| row 2, cell 2
| row 2, cell 3
|}
```

produces a table that looks like this:

Header 1	Header 2	Header 3
row 1, cell 1	row 1, cell 2	row 1, cell 3
row 1, cell 1	row 1, cell 2	row 1, cell 3

Notice how we don't explicitly make the header row bold. We simply indicate in the **structure** that those cells are headers by using `!` at the start of the cell's line instead of the normal `|`. Indigo will format the cell appropriately.

Telling Indigo to Ignore Special Items

Sometimes it's useful to be able to tell Indigo not to interpret a line specially. Do this by starting the line with a backslash `\`.

This is particularly useful when a paragraph starts with a text that Indigo would normally interpret as a section or subsection. For example:

```
Fill in this form:

\1. State ....................................
\2. Date Received ........................
\3. Checked ..............................
\4. Certificate Issued ...................
\5. Certificate No. ......................
\6. Treasury Receipt No. .................
```

By starting each line with `\` Indigo knows that it shouldn't treat those lines as section numbers.

Adding new Chapters, Parts and Sections

You can easily add a new chapter, part or section to a document. To do so:

1. edit the chapter, part or section just before where the new one needs to go
2. at the bottom, add the new chapter, part or section heading and content

3. click the green Update button

Links

Add a link in the text of your document using this syntax:

```
[link text](http://example.com/page)
```

That will create a link like this: link text

Images

You can embed an image in your document using this syntax:

```
![alternative text](/media/image.png)
```

That will create an image using the `image.png` file added to your document as an attachment.

Downloading PDF and Standalone HTML

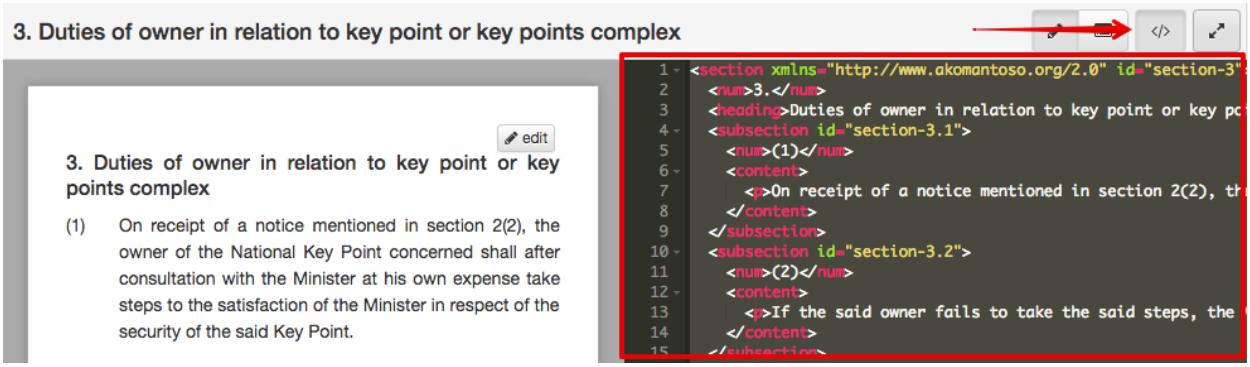
You can download PDF and standalone (self-contained) HTML versions of a document. These are useful for distribution and archiving. Go to **Preview** and click the **Download** button.

Downloaded and PDF documents can have a *colophon*, which is a page right at the start of the document which contains copyright, ownership and other information. A colophon is automatically chosen for a document based on document's country.

Note: Administrators can add new colophons through the Admin interface. Click on your name in the top-right corner and choose **Site Settings**.

Viewing the XML

It can be useful to see what the Akoma Ntoso for a piece of the document looks like. Click the **Show Code** button to do this:



1.2.6 The Admin Area

Indigo has a backend Admin area that lets adminstrator users control some internal settings of how Indigo works. Only adminstrator (staff) users have access to this area.

Logging In

To log into the Admin area:

1. Log into Indigo
2. Click your name in the top-right corner
3. Click **Site Settings**

Note: If the **Site Settings** option isn't visible, then you aren't a staff member and don't have permission to view the admin area.

Permissions

Indigo uses Django's permission system to control who can create, edit, delete and publish documents. This can help prevent documents from being deleted and published accidentally.

Permissions can be assigned to individual users, or to **groups**. Users who belong to groups get permissions from those groups.

To edit permissions,

1. Click on **Users** or **Groups** in the Admin area.
2. Choose the user or group to edit (or create a new user or group)
3. Scroll down to the permissions boxes.

The permissions that are important are:

indigo_api | work | Can add work Allows the user to create new works. This doesn't allow a user to edit existing works.

indigo_api | work | Can change work Allows the user to edit existing works.

indigo_api | work | Can delete work Allows the user to delete works.

Note: Only give work permissions to experienced users.

---

indigo_api | document | Can add document Allows the user to create and import a new document. This doesn't allow a user to edit existing documents.

indigo_api | document | Can change document Allows the user to edit existing documents.

indigo_api | document | Can delete document Allows the user to delete documents.

indigo_api | document | Can publish and edit non-draft documents Allows the user to mark a document as *published (not a draft)* and, along with the *change* permission, edit published documents.

Note: Only give the **delete** and **publish** permissions to experienced users.

---

indigo_api | annotation | Can add annotations Allows the user to annotate (add comments) to documents.

indigo_api | annotation | Can change document Allows the user to edit their annotations.

indigo_api | annotation | Can delete annotation Allows the user to delete their annotations.

Adding New Admins

To give admin permissions to a user, which allows them to log into the Admin area:

1. Click on **Users** in the Admin area.
2. Click on the user you wish to make an Admin.
3. Check the **Staff status** checkbox in the Permissions section and click Save.

1.3 Introduction to the Terminology of Law

What do you understand by the word "law"? It is a word which can have many meanings, but in the language of law publishing, it means a piece of legislation which regulates the way in which things are done or should be done in many spheres of life, especially in public, in order to maintain good order and prevent the growth and flourishing of anarchy.

Law according to the Roman Dutch and British systems has developed over several centuries. A great deal of the language is therefore antiquated and sometimes quite difficult to understand.

Note: This content is adapted with permission from a training guide prepared by Adrienne Pretorius for AfricanLII.

1.3.1 Levels of legislation

In South Africa, we have three levels of legislation. These are **national**, **provincial**, and **municipal**. Most jurisdictions have at least two levels: national and local government.

National legislation: This is usually published as Acts and Regulations and/or Rules. See the later sections on publication of an Act.

Provincial legislation: When our nine provinces were established, they already had a number of Ordinances which authorised them to make bylaws at municipal level. In addition to being allowed to retain many of their Ordinances when the changes were made from four to nine provinces, they were allocated certain levels of legislative involvement in respect of which they were granted the power to draft, debate on, pass and eventually publish Acts and Regulations and/or Rules. The old Ordinances were gradually repealed, and very few of them are still in existence today. When provincial legislation originally got under way in the early 1990s, the early provincial Acts used to be called laws, but they are now all called Acts.

Municipal legislation: This is usually published as Bylaws.

As most jurisdictions publish Acts, Regulations and/or Rules, we are going to concentrate on this level of legislation.

Note: From here onwards, any reference to an Act or Acts also applies to other types of legislation. You will learn about some of these later.

1.3.2 Publication of Acts

A *Gazette* is an official newspaper, and in most countries, legislation as well as a large variety of other legal notifications are published in gazettes. In South Africa, Acts are usually published by means of a Government Notice in *Gazettes*. There are some exceptions. For example, our Public Service Act of 1994 does not have an Act number; it is Proclamation 103 of 1994. This is very unusual in terms of our legal system, which falls into a classification known as Roman-Dutch law. We follow this system because the first colonial settlers were Dutch, and they introduced the same system that they used in their home country of Holland.

A *Gazette* has a number of important features:

- Firstly, the **type of Gazette** (national or provincial) is shown clearly on the cover (first page) of the Gazette. Either our national coat of arms or the coat of arms of one of our nine provinces will appear on that page, together with the words *"National Gazette"* or *"Provincial Gazette"*.
- Secondly, the **date** must appear on the first page. This is extremely important. Do you know why? The reason is that the date of commencement of an Act is often linked to the date of publication.
- Thirdly, there will be a **Gazette number**. This information helps us when we are searching for missing legislation.
- Below those details will be a short paragraph telling us who **assented** to the Act (agreed that it could be published and made available to the public). This section will include the **Government Notice (or Provincial Notice) number** and the **date** on which the Act is promulgated. This is also sometimes referred to as the date on which the Act was signed into law.
- This is followed by the **number** and **year** of the Act, as well as its **short title**.

Note: IT IS IMPORTANT TO NOTE THAT THE FRONT COVER DOES NOT FORM PART OF THE ACT, AND THAT THERE ARE OFTEN ERRORS ON THIS PAGE (THE COVER). **NEVER** TAKE THE NUMBER, YEAR OR TITLE OF AN ACT FROM THIS PAGE. YOU **MUST** CHECK AGAINST THE DETAILS INSIDE THE GAZETTE AND IN THE SHORT TITLE OF THE PUBLISHED ACT.

Other information on the first page will usually include the volume number of the *Gazette* and the place in which the *Gazette* was printed. In the case of South African Acts, this is often Cape Town, but most of our *Gazettes* are printed in Pretoria (Tshwane), where the official premises of the Government Printer are situated.

1.3.3 Types of Act

Acts fall into a number of categories. The two main categories are principal and amendment Acts. Others include Appropriation Acts and Division of Revenue Acts, which regulate the spending of public money (these are said to be **ephemeral** – the information which they contain changes on a regular basis).

Principal Acts

Principal Acts are by far the most important category of Act. They lay the foundation for court decisions.

Note: Always remember that Statute law and judicial precedent (Court judgments) form the foundation of any justice system.

Amendment Acts and other amending notices

It is important to note that it does not matter how many times a principal Act is amended; it will ALWAYS retain its original number (for example, Act 71 of 2008). The only time the Act number may change is in a case where an error has been made in the Statutes books and the Act has inadvertently been incorrectly numbered. I can think of only one instance where this has happened, and it was long before I became a Statutes editor. It is far more likely that the short title will be changed altogether (in other words, the Act is renamed).

In most countries, principal legislation may be amended by principal Acts, amendment Acts, and subsidiary legislation. Some Acts do not provide for this, or provide specifically for amendment only by Act. Remember that principal Acts may also contain at least one repeal, if the Act replaces an existing Act dealing with the same area of law.

Amendments of any kind cannot be made if the amending provisions are not in operation. Always check before you start updating, and also check for whether certain provisions commence on different dates from the rest of the Act.

It is also important to know that amendment Acts do not always amend only one Act. The short title may therefore not be a guide as to which Act is amended. For example, a Revenue Laws Amendment Act may amend several Acts or every Act dealing with revenue (State income), or only one or two, and it may also amend other Acts.

1.3.4 Components (parts) of an Act

A law or Act is usually made up of a number of different components or parts. Some of these are:

- The title, usually called the "short title", which is the name by which the Act is known. The short title often has a reference to the date of commencement (see the later section dealing with dates of commencement)
- Details with regard to the date of promulgation (publication); who assented to (approved) the legislation; which language was signed (in multilingual countries/jurisdictions); and what the date of commencement is
- The long title, which explains briefly why the Act was made
- An Arrangement of Sections (this is not part of the law, but assists in navigating through an Act or other piece of legislation, especially lengthy documents)
- Chapters or Parts, identified by their headings (these are subdivisions used to arrange the information in an Act or other piece of legislation. In some countries, especially those which were once under British rule, the Acts are known as Chapters or by a Latin abbreviation, "Cap")
- A number of sections/regulations/rules (according to the nature of the legislation)
- Side headings to the various sections (as with the Arrangement of Sections, these are not part of the law, but assist in navigating through an Act or other piece of legislation)
- A section containing definitions (the meaning of words as used in the particular Act. This is a very important section, because words have different meanings in different contexts; and in law, we very often have to look at other Acts to see what a certain word means in context)
- Subsections/subregulations/subrules (as above)
- Paragraphs (usually (a), (b), and so on)
- Subparagraphs (usually identified by small Roman numerals: (i), (ii), (iii), and so on)
- Schedule(s)
- Tables and diagrams.

The *short title* is the commonly known name of a law or Act, as it is called in most countries. It usually includes the number of the Act and the year in which it belongs. Many years ago, Acts were always passed in the year to which they were allocated. For example, all Acts passed in 1984 were printed and published before the end of 1984. As

the years have gone by, parliamentary sessions have become longer and less orderly, and as a result, less legislation is passed in each year.

However, Acts which have not been promulgated and published in the year in which they are allocated will be passed in a subsequent year, **but will still retain the original year in the number**. For example, we are still waiting for the Protection of State Information Act 41 of 2013 (known commonly in South Africa as the "Secrecy" Act) to be published. The number has been allocated, but the Act has not yet been promulgated. Even if it is published only in five years' time, it will still be Act 41 of 2013. In some cases, Acts which are awaiting promulgation are repealed (removed from the Statutes book) without ever being made available to the public except in Bill form.

Note:

- A reference to "**the Statutes book**" refers nowadays to the computerised record of all principal and amendment Acts published and still operative in terms of South African law. Many years ago, this was a physical book (or rather, a very large set of books) containing these records, and a repeal entailed physically removing the record of the particular Act from the official book.
- A **Bill** is a preview of what an Act will contain. Some Bills are passed after their first version has been published, while others may go back time and again for further revision by the various Parliamentary committees.]

The **long title** is a summarised breakdown of what the purpose of the Act is. It may be presented in only a few words, in some cases, while in others, particularly financial measures, it can run to several pages.

The **Arrangement of Sections/Regulations/Rules** is an aid to navigating through a document, especially a very long one. It does not form part of the law, and in fact, many Acts have no Arrangement of Sections. If you are ever involved in law revision and consolidation, you will probably be asked to insert an arrangement of Sections wherever there isn't one.

Chapters and **Parts** are subdivisions in South African Acts. There is no set pattern as to how these are applied in an Act. Sometimes there are Chapters only, and sometimes Parts only. In some cases, Chapters are divided into Parts. So there is no hard-and-fast rule regarding the use of Chapters and Parts in South African legislation. In some countries, a reference to a "Chapter" is a reference to an Act, and is usually abbreviated to "Cap", as mentioned above.

Sections are the "meat" of an Act. They flesh out the provisions relating to the subject matter of an Act. They are numbered in sequence, but insertions are often made at a later stage, modifying the original numbering section by inserting, for example, a new section 13A.

The **side headings** to sections (also referred to sometimes as "marginal notes") do NOT form part of the law, and can be edited and/or replaced when revising and consolidating legislation. In an ordinary revision service, it is not wise to change these side headings, because people tend to refer to a particular section by that side heading as well as the number of the section.

Paragraphs, **subparagraphs** and **items** are further subdivisions in an Act (and are sometimes even further subdivided), but they depend on a higher level of the hierarchy of the Act (see the section below on the hierarchy of the components of an Act) – they will usually not be found standing independently in an Act.

We refer later to "lists" of legislation in Acts. This means any sequential arrangement of information, so it could refer to the sections, paragraphs, and so on, as well as what are generally known as "lists" (see the section below on the hierarchy of the components of an Act, where there is a list of words).

Schedules fall into many different categories. The majority of them will contain amendments and/or repeals of legislation. However, many of them contain provisions supplementary to the text of the Act in which they appear. For example, an Act dealing with the establishment of a new council may contain the constitution, functions and duties of the council and its members. The South African Labour Relations Act 66 of 1995 contains a Code of Dismissal in one of its numerous Schedules. Acts relating to the Defence Force or Police or other protective bodies often have their regulations in a Schedule following the text of the Act. The South African Criminal Procedure Act 51 of 1977 has schedules relating to various categories of crime according to how severely these offences are punishable.

Whatever form it may take, a Schedule to an Act always contains supplementary information of importance in meeting the objectives of the Act. There is usually an indication of the section which governs the Schedule, and by referring to that section number, you will be able to see what the purpose of the Schedule is. A Schedule may also be governed by a number of sections.

Tables and **diagrams** can cause problems. It may not always be possible to position insertions of this kind exactly where they appear in the Gazette. For this reason it is advisable not to refer to tables, illustrations or diagrams by page number or by phrases such as "the diagram below". Each individual table, illustration or diagram should be given a specific label.

1.3.5 Hierarchy of various features forming part of an Act

The words below show the parts into which an Act is usually divided, and how we name those parts:

- Chapter/Part
- Section
- Subsection
- Paragraph
- Subparagraph
- Item
- Subitem (or sub-item)
- Sub-subitem (or sub-sub-item)
- Schedule

We usually abbreviate the terms above when annotating. Always check Acts already in the database before you start annotating to ensure that your abbreviations match those already in the publication, if abbreviations have been used.

1.3.6 Cut-off date

When doing any type of law revision, it is important to decide exactly which legislation is to be included in the update. Once you have collected all the new legislation, go through all of the notices carefully and arrange them in chronological order. The date of the last notice which is in operation should be set as the cut-off date, unless you work specifically within monthly, quarterly or half-yearly periods. It is good practice not to change the date continually, but it also pays to be flexible sometimes. If a really important piece of legislation is passed after the cut-off date, the value to your subscribers/clients of including that notice should be weighed against working strictly to schedule and excluding legislation published after the cut-off date.

1.3.7 Identifying under which title an Act belongs

It is useful to classify published legislation under a number of different titles rather than searching through a large database with no subdivisions. This is usually done by seeing which Acts are controlled by the same Minister (for example, the Minister of Agriculture). Most Acts contain a definition indicating which government department or Minister (or MEC or other person or organisation) controls the provisions of that Act. If there is no specific Minister or department named, it is sometimes possible to find this information by knowing where to look for clues. The first page of Gazettes containing Acts will usually indicate who has assented to the Act. While this is usually the head of the government, it may in fact be the relevant Minister, provided that the Act allows for that contingency. Nowadays, various government websites contain legislation relating to particular portfolios (areas of responsibility). So a search on Google will often provide the answer you need. If necessary, or when you are unsure, you should contact the legal

drafting section of your Legislature and ask them for clarity. This applies only if you classify your information as outlined above. Some countries prefer not to do so.

1.3.8 Determining the date of commencement

No amendments can be made to legislation if the amending provisions are not in operation.

When no indication of the date of commencement is given in the short title or a section dealing exclusively with commencement, the date of commencement is usually the date of the Gazette.

However, **THIS IS NOT ALWAYS THE CASE**.

Before you enter a date or dates of commencement, you need to consult the **Interpretation Act** of the particular country or jurisdiction. In most cases, it is made clear that the date of commencement is the date of the Gazette if no other indication is made. But sometimes, the date of commencement is defined differently. For example, section 13 of our Interpretation Act 33 of 1957 (commencement of laws) provides as follows:

> "(1) The expression 'commencement' when used in any law and with reference thereto, means the day on which that law comes or came into operation, and that day shall, subject to the provisions of subsection (2) and unless some other day is fixed by or under the law for the coming into operation thereof, be the day when the law was first published in the Gazette as a law.
>
> "(2) Where any law, or any order, warrant, scheme, letters patent, rules, regulations or by-laws made, granted or issued under the authority of a law, is expressed to come into operation on a particular day, it shall be construed as coming into operation immediately on the expiration of the previous day."

Note: In other words, if the date of commencement is 18 May 2015, the piece of legislation will commence as the clock strikes midnight on the night of 17 May 2015.

> "(3) If any Act provides that that Act shall come into operation on a date fixed by the President or the Premier of a province by proclamation in the Gazette, it shall be deemed that different dates may be so fixed in respect of different provisions of that Act."

The word "deemed" means "considered" or "understood".

Note that there may be many different dates of commencement in any given Act. If this is done, the short title or the provision regarding commencement should allow for different dates to be applied. However, the Interpretation Act has provided for cases where the draftspersons have not included this provision in an Act (see subsection (3) in the quotation above).

A date of commencement may be applied retrospectively. This means that a date of commencement may be earlier than the date of the Gazette. This is done for various reasons, which are not important at this point.

Meaning of "promulgation" and "proclamation"

The word "**promulgation**" means making known to the public. A word which is often used in the same sense is "**proclamation**". However, in law, promulgation of an Act must take place before a proclamation made under that Act can be issued. The new Act may make provision for a proclamation to be made determining the date or further dates of commencement. Sometimes a "notice" is referred to. In South Africa this is usually a Government Notice.

The date on which an Act commences is not necessarily the first date which is proclaimed. While that date certainly applies, if it brings only a few provisions into operation, the actual date of commencement will be one when most of the provisions not yet in operation are brought into force. In cases like this, you will use the first date of commencement, until it becomes clear that that date is not appropriate. You will then need to insert the correct date of commencement, and add the words "(unless otherwise indicated)", then annotate the sections which came into operation earlier (see

the section on annotating). Do you see above how many different expressions can be used in connection with dates of commencement? You'll eventually become familiar with all of them, even if it seems confusing at first.

1.3.9 Consolidating and annotating

This kind of work involves two very important concepts: **consolidating** and **annotating**.

If you are busy with a law revision, consolidation is a very large part of what you have to do. It is accompanied by annotation, the purpose of which is to ensure that the history of what has happened to a particular piece of legislation is shown fully. Annotations are usually added below the part of an Act which is amended, although there are exceptions. As they are interventions by the editor (they are not part of the original text), they should be in square brackets.

Consolidation of legislation is usually done to simplify legislation which has been frequently amended. All superfluous or obsolete provisions are removed, and the Legislature and the Courts will now have a relatively "clean" piece of legislation with which to work, showing all changes which have occurred to the Act since it was first published except for the ones which are no longer of value in the eyes of the law or which had previously been repealed.

As you will probably not be involved in consolidating legislation for a law revision (which is usually done by experts in the field), we won't spend too much time on that right now. We'll look at updating (which is part of consolidation) and annotating.

1.3.10 Inserting/Adding

You will often be asked to add or insert provisions, and it is important to differentiate between the two functions. Even legal draftspersons occasionally confuse them, so be very careful when reading instructions in amending legislation. Inserted provisions are usually quite easy to identify. They will most often have a capital A, B, and so on after the section or subsection (or paragraph) number (for example, "section 5A", or "subparagraph (ivD)", or other similar expressions). However, you may have a section in an amendment Act which requires you to insert provisions, but these are purely and simply subsequent numbering, for example, "insert paragraphs (d) and (e) after paragraph (c)". It is quite clear that these paragraphs are not being inserted; they are being added, and you can reflect that in your annotations:

> [Para (d) **added** by . . .]

Similarly, if the instruction requires you to add subsection (3A), it is more likely that (3A) is being inserted:

> [Subsection (3A) **inserted** by . . .]

You may wonder why we are allowed to change these words to suit what is actually done instead of following what is in the Gazette. The reason we can do so is that they are in the editor's annotations, which do not form part of the law.

Occasionally, you will find that something had previously been deleted (repealed), and you are re-inserting it. It does happen from time to time, and you need to annotate it exactly as if it is a brand new insertion.

You may also occasionally find that you are asked to delete or repeal something which has previously been annotated as having been deleted or repealed. In a case like that, simply update the annotation. If you put in a date of commencement, it will make what has happened clearer to subscribers and clients, but that kind of enhancement must be part of your updating style – it must be applied consistently.

If you feel uneasy about that remedy, contact legal drafting and discuss the problem with them.

Changing of punctuation after adding or inserting

Sometimes you will be asked to add something at the end of an existing series of entries. For example, if you look at the instruction to "insert paragraphs (d) and (e) after paragraph (c)" (you will actually be adding them), the chances are that (c) has a full stop at the end. If it does, you will need to change that to a semi-colon or comma (more often

a semi-colon – check what has been used previously in the listed series), add (not insert) and annotate the two new paragraphs, and ensure that the last one in the series has the same punctuation as (c) originally had.

1.3.11 LRO (Law Revision Order) or Revision Service number

If your legislation is printed, only parts which have changed will be updated (see the section on loose-leaf updating – loose-leaf is the most cost-effective way of publishing a print update). The Attorney-General or President/Prime Minister or head of the legislative drafting services will issue a notice at some point which will proclaim or determine the cut-off date, or mention that the legislation has been updated from the day after the previous service up to the cut-off date for the current service. This notice is usually referred to as the LRO for the update, and the LRO number will be the identifying number for the service. It must be reflected on all pages which have been replaced during the update. In countries where the LRO system is not used (for example, South Africa), this number will be referred to as the Revision Service (or merely Service) number.

1.4 General API Guidelines

This guide is for developers who want to use either the Indigo Public REST API or the Indigo Application REST API. We assume that you have a basic understanding of web applications, REST APIs and the Akoma Ntoso standard for legislation (acts).

This guide covers details that are shared by both APIs. For more information on each API, see *Using the Application REST API* and *Using the Public REST API*.

1.4.1 Location of the API

The API is available at http://indigo.code4sa.org/api/.

It is easy to explore using a browser and follows normal REST semantics using HTTP methods such as `GET`, `POST`, `PUT` and `DELETE`.

1.4.2 Content types

Some REST calls can return content in multiple formats. You can specify the content type of a response by placing `.format` at the end of the URL or including an `Accept:` header in the request. In most cases the default response type is JSON.

- `.json` or `Accept: application/json`: return JSON
- `.xml` or `Accept: application/xml`: return Akoma Ntoso XML
- `.html` or `Accept: text/html`: return human friendly HTML
- `.epub` or `Accept: application/epub+zip`: return an ePUB (ebook) document
- `.pdf` or `Accept: application/pdf`: return a PDF document
- `.zip` or `Accept: application/zip`: return a ZIP file with the document XML and media attachments

Note: Not all responses support all formats, the documentation will be explicit about what is supported.

When submitting data to the API in a PUT or POST request, encode it either as JSON or using form encoding and include the appropriate `Content-Type` header.

An example POST request with JSON encoded values:

```
POST /api HTTP/1.1
Accept: application/json
Content-Type: application/json; charset=utf-8

{
    "key": "value"
}
```

An example POST request with form encoded values:

```
POST /auth/login/ HTTP/1.1
Accept: application/json
Content-Type: application/x-www-form-urlencoded; charset=utf-8

key=value
```

All our examples will use the JSON format for requests.

See also:

For more information on content type negotation see http://www.django-rest-framework.org/api-guide/content-negotiation/

1.4.3 Document (Act) Details

Both APIs describe a document (an act) as a JSON object, such as:


```
{
  "amended_versions": [{
    "id": 2,
    "expression_date": "2005-03-01"
  }],
  "amendments": [{
    "date": "2005-03-01",
    "amending_title": "Act to Amend Act 10 of 2004",
    "amending_uri": "/za/act/2005/1",
  }],
  "assent_date": "2004-03-03",
  "content_url": "http://indigo.code4sa.org/api/documents/1/content",
  "country": "za",
  "created_at": "2015-01-14T15:57:08.497844Z",
  "draft": false,
  "frbr_uri": "/za/act/2004/10/eng",
  "expression_date": "2004-05-21",
  "commencement_date": "2004-05-21",
  "id": 1,
  "language": "eng",
  "locality": null,
  "nature": "act",
  "number": "10",
  "publication_date": "2004-05-21",
  "publication_name": "Government Gazette",
  "publication_number": "179",
  "published_url": "http://indigo.code4sa.org/api/za/act/2004/10/",
  "stub": false,
```


(continues on next page)

(continued from previous page)


```
  "subtype": null,
  "tags": ["checks needed"],
  "title": "Act 10 of 2004",
  "updated_at": "2015-02-17T12:23:48.394662Z",
  "url": "http://indigo.code4sa.org/api/documents/1.json",
  "year": "2004"
}
```


Each of these fields is described in the table below.

Field	Description	Type	Default for new documents
amendments	List of amendments that have been applied to create this version of the document. Read-only.	See below	`[]`
amended_versions	List of different amended versions of this document in the library. Read-only.	See below	`[]`
assent_date	Date when the document was assented to. Read-only.	ISO8601	
content_url	URL of the full content of the document. Read-only.	URL	Auto-generated
country	ISO 3166-1 alpha-2 country code that this document is applicable to.	String	
created_at	Timestamp of when the document was first created. Read-only.	ISO8601	Current time
draft	Is this a draft document or is it available in the public API?	Boolean	`true`
expression_date	Date of this expression (or publication). Required.	ISO8601	Publication date
commencement_date	Date of this commencement of most of the document. Read-only.	ISO8601	
frbr_uri	FRBR URI for this document.	String	None, a value must be provided
id	Unique ID of this document. Read-only.	Integer	Auto-generated
language	Three letter ISO-639-2 language code for the language of the document.	String	`"eng"`
locality	The code of the locality within the country. Optional. Read-only.	String	
nature	The nature of this document, normally “act”.	String	`"act"`
number	Number of this act in its year of publication, or some other unique way of identifying it within the year	String	
published_url	URL of where the published document is available. This will be null if draft is true	URL	Auto-generated
stub	Is this a stub document? Stub documents are generally empty.	Boolean	`false`
subtype	Subtype code of the document. Optional. Read-only.	String	
tags	List of string tags linked to the document. Optional.	Strings	`[]`
title	Document short title.	String	`"(untitled)"`
updated_at	Timestamp of when the document was last updated. Read-only.	ISO8601	Current time
url	URL for fetching details of this document. Read-only.	URL	Auto-generated
year	Year of publication	String	

In some cases, a document may also contain a `content` field.

Field	Description	Type	Default for new documents
content	Raw XML content of the entire document.	String	Basic document content

1.4.4 Amendments

Amendments describe works that made amendments to this document.

Field	Description	Type
amending_title	Title of the amending document.	String
amending_uri	FRBR URI of the amending document.	String
date	Date of the amending document, the date at which the amendment took place.	ISO8601

1.4.5 Amended Versions

The amended versions are those documents in the library with the same FRBR URI and different expression dates. They are looked up automatically for a document, so it is important that the FRBR URI and expression date are correct for all documents. All fields are read-only.

Field	Description	Type
id	Document id in the library.	Integer
expression_date	Date of the expression (or publication) of the document.	ISO8601

1.4.6 Pagination

APIs that list items, such as document and attachment lists, will be paginated and return a limited number of items per page. The response includes information on the number of total items and the URLs for the next and previous pages.

Here's an example with 250 total items and two pages:

```
{
  "count": 250,
  "next": "http://indigo.code4sa.org/api/documents.json?page=2",
  "previous": null,
  "results": [ "..." ]
}
```

In this case, fetching the `next` URL will return the second (and final) page.

1.4.7 Next Steps

Now you're ready to read the guides for the two APIs:

- *Using the Application REST API*
- *Using the Public REST API*

1.5 Using the Public REST API

This guide is for developers who want to use the Indigo Public REST API to fetch and render documents for users to read or download. We assume that you have a basic understanding of web applications, REST APIs and the Akoma Ntoso standard for legislation (acts).

See *General API Guidelines* for general API details such as content types and what the fields of a Document are.

If you want to manage and edit a collection of legislation see *Using the Application REST API* instead.

1.5.1 Public API

The public API is a read-only API for exploring a collection of legislative documents. Using it, you can:

- get a list of all acts by country and year
- get the raw Akoma Ntoso XML of an act
- get an human-friendly HTML version of an act
- get an Atom feed of newly published acts

The public API relies heavily on FRBR URIs (and URI fragments) for identifying content, be sure to read up on FRBR URIs above.

Note: When we use a URL such as `/api/frbr-uri/` in this guide, the `frbr-uri` part is a full FRBR URI, such as `/za/act/1998/84/eng`.

1.5.2 Listing Acts

```
GET /api/za/
GET /api/za/act/
GET /api/za/act/2007/
```

- Content types: JSON, PDF, EPUB, ZIP

These endpoints list all acts for a country or year. To list the available acts for a country you'll need the two-letter country code for the country.

1.5.3 Atom Feeds

```
GET /api/za/summary.atom
GET /api/za/full.atom
GET /api/za/act/summary.atom
GET /api/za/act/full.atom
GET /api/za/act/2007/summary.atom
GET /api/za/act/2007/full.atom
```

- Content types: Atom

There are two Atom feeds for documents: a summary feed (`summary.atom`) and a full content feed (`full.atom`). The summary feed has the act title, metadata and the act preface (if any) of each document. The full content feed has the full HTML content of each document.

The Atom feeds are paginated and contain the most recently updated documents first. The summary feeds contains more items per page than the full feeds because the latter are much larger. Follow the `next` links in the feeds to fetch additional pages.

1.5.4 Entire Act

```
GET /api/frbr-uri
```

- Parameter `coverpage`: should the response contain a generated coverpage? Use 1 for true, anything else for false. Default: 1. (HTML-only)

- Parameter `standalone`: should the response by a full HTML document, including CSS, that can stand on its own? Use 1 for true, anything else for false. Default: false. (HTML-only)
- Parameter `resolver`: the fully-qualified URL to use when resolving absolute references to other Akoma Ntoso documents. Use 'no' or 'none' to disable. Default is to use the Indigo resolver. (HTML-only)
- Content types: JSON, XML, HTML, PDF, ePUB, ZIP

This returns the entire contents of an act. For example, the English HTML version of `/za/act/1998/84/eng` is available at:

```
http://indigo.code4sa.org/api/za/act/1998/eng.html
```

The raw XML is available at:

```
http://indigo.code4sa.org/api/za/act/1998/eng.xml
```

1.5.5 Acts at a Point in Time

A URI such as `/za/act/1998/84/eng` actually refers to the latest (current) version of the act.

An act may be amended multiple times over its lifetime. You can retrieve the version of an act as it appeared after a dated amendment, if available, by specifyng the date in the URI in the format `@YYYY-MM-dd`. For example, `/za/act/1998/84/eng@2012-01-01` is the version of Act 84 of 1998 after the amendment on date 2012-01-01 has been applied. If there was no amendment of that document on that exact date, a 404 will be returned.

You can fetch the very first version of the act by using a `@` without a date: `/za/act/1998/84/eng@`.

If you don't know on which exact dates amendments were made, you can get the version of the act as it would have looked on a particular date (if available) by placing `:YYYY-MM-DD` at the end of the URI, for example: `/za/act/1998/84/eng:2012-06-01`. Indigo will find the most recent amended version at or before that date.

Components and formats are placed after the date portion, such as `/za/act/1998/84/eng@2012-01-01.json`.

1.5.6 Table of Contents

```
GET /api/frbr-uri/toc.json
```

- Content types: JSON

Get a description of the table of contents (TOC) of an act. This includes the chapters, parts, sections and schedules that make up the act, based on the structure captured by the Indigo editor.

Each item in the table of contents has this structure:

```
{
  "id": "chapter-1",
  "type": "chapter",
  "num": "1",
  "heading": "Interpretation",
  "title": "Chapter 1 - Interpretation",
  "component": "main",
  "subcomponent": "chapter/1",
  "url": "http://indigo.code4sa.org/api/za/act/1998/10/eng/main/chapter/1",
  "children": [ "..." ]
}
```

Each of these fields is described in the table below.

Field	Description	Type
id	The unique XML element id of this item. (optional)	String
type	The Akoma Ntoso element name of this item.	String
num	The number of this item, such as a chapter, part or section number. (optional)	String
heading	The heading of this item (optional)	String
title	A derived, friendly title of this item, taking `num` and `heading` into account and providing good defaults if either of those is missing.	String
component	The component of the Akoma Ntoso document that this item is a part of, such as `main` for the main document, or `schedule1` for the first schedule.	String
subcomponent	The subcomponent of the component that this item is a part of, such as a chapter. (optional)	String
url	The API URL for this item, which can be used to fetch XML, HTML and other details of just this part of the document.	String
children	A possibly-empty array of TOC items that are children of this item.	Array

1.5.7 Fetching Parts, Chapters and Sections

You can use the `url` field from an item in the table of contents to fetch the details of just that item in various forms.

```
GET /api/frbr-uri/toc-item-uri.format
```

- Content types: XML, HTML, PDF, ePUB, ZIP

1.5.8 Using HTML Responses

Indigo transforms Akoma Ntoso XML into HTML5 content that looks best when styled with Indigo Web stylesheets. You can link to the stylesheets provided by that package, or you can pull them into your website.

1.5.9 Search

```
GET /api/search/works?q=<search-term>
```

- Parameter `q`: the search string
- Filter parameters:
  - `country`
  - `draft`
  - `frbr_uri`, `frbr_uri__startswith`
  - `language`
  - `stub`
  - `expression_date`, `expression_date__lte`, `expression_date__gte`
- Content types: JSON

This API searches through works (acts). It returns all works that match the search term, in search rank order. Each result also has a numeric `_rank` and an HTML `_snippet` with highlighted results.

Use additional parameters to filter the search results.

If more than one expression of a particular work matches the search, then only the most recent matching expression is returned. If you would like all matching documents, use the `/api/search/documents` search API.

1.6 Using the Application REST API

This guide is for developers who want to use the Indigo REST Application API to manage and edit works and documents. We assume that you have a basic understanding of web applications, REST APIs and the Akoma Ntoso standard for legislation (acts).

See *General API Guidelines* for general API details such as content types and what the fields of a Document are.

If you only want to fetch and render published legislation and don't care about works or editing legislation, see *Using the Public REST API* instead.

1.6.1 Application API

The application API is the REST API used by the web editor to manage documents. With it you can:

- create, edit, update and delete works
- list works
- create, edit, update and delete documents
- list and search for documents
- manage users and passwords

The API is available at http://indigo.code4sa.org/api/.

It is easy to explore using a browser and follows normal REST semantics using HTTP methods such as `GET`, `POST`, `PUT` and `DELETE`.

1.6.2 Authentication

Write operations (POST, PUT and DELETE) require authentication with a token. To get an authentication token, use

- `POST /auth/login/`
  - `username`: your email address
  - `password`: your password

and store the returned `key` as your token.

```
POST /auth/login/ HTTP/1.1
Accept: application/json
Accept-Encoding: gzip, deflate
Content-Length: 56
Content-Type: application/json; charset=utf-8

{
    "password": "password",
```

(continues on next page)

(continued from previous page)

```
    "username": "me@example.com",
}
```

```
HTTP/1.0 200 OK
Allow: POST, OPTIONS
Connection: close
Content-Type: application/json

{
    "key": "118365019bd8a541e9211dc12741c927225ec00a"
}
```

In subsequent requests that require authentication, include the token as a header `Authorization: Token 118365019bd8a541e9211dc12741c927225ec00a`.

See also:

For more information on token authentication, see the authentication documentation for the Django Rest Framework.

1.6.3 Works

A Work is an Act, a by-law, a regulation, and so on. In Indigo, a work doesn't have any content – it's just a description of the basic details of the act (or by-law, etc.). A work can be associated with many documents, each of which is an *expression* of the work.

An Indigo work is uniquely identified by its FRBR URI. Documents are linked to a work through a common FRBR URI.

Documents inherit a number of fields from a work, such as the FRBR URI, publication date, repeal status, etc.

List Works

```
GET /api/works
```

Lists the works. The results will be *paginated*.

Get a Work

```
GET /api/works/{id}
```

Fetches a JSON description of a work. For example:

```
{
  "assent_date": null,
  "commencement_date": "2018-04-11",
  "country": "za",
  "created_at": "2018-04-07T11:59:28.181610Z",
  "created_by_user": {
    "id": 1,
    "display_name": "Greg K."
  },
  "frbr_uri": "/za/act/2018/2",
  "id": 246,
```

(continues on next page)

(continued from previous page)


```
  "locality": null,
  "nature": "act",
  "number": "2",
  "publication_date": "2018-04-11",
  "publication_name": "Government Gazette",
  "publication_number": 1234,
  "repealed_by": null,
  "repealed_date": null,
  "subtype": null,
  "title": "An Act",
  "updated_at": "2018-04-07T11:59:28.181651Z",
  "updated_by_user": {
    "id": 1,
    "display_name": "Greg K."
  },
  "url": "http://indigo.code4sa.org/api/works/246",
  "year": "2018"
}
```


Each of these fields is described in the table below.

Field	Description	Type	Default for new works
assent_date	Date when the work was assented to. Optional.	ISO8601	
commencement_date	Date of this commencement of most of the work. Optional.	ISO8601	
commencing_work	Work that determined the commencement date, if any. Optional.	Object	
country	ISO 3166-1 alpha-2 country code that this work is applicable to.	String	
created_at	Timestamp of when the work was first created. Read-only.	ISO8601	Current time
frbr_uri	FRBR URI for this work.	String	None, a value must be provided
id	Unique ID of this work. Read-only.	Integer	None, a value must be provided
locality	The code of the locality within the country. Optional. Read-only.	String	
nature	The nature of this work, normally "act".	String	`"act"`
number	Number of this act in its year of publication, or some other unique way of identifying it within the year	String	
repealed_by	Work that repealed this work, if any. Optional.	Object	
repealed_date	Date when this work was repealed. Optional.	ISO8601	
subtype	Subtype code of the work. Optional. Read-only.	String	
title	work short title.	String	None, a value must be provided
updated_at	Timestamp of when the work was last updated. Read-only.	ISO8601	Current time
url	URL for fetching details of this work. Read-only.	URL	Auto-generated
year	Year of publication	String	

Update a Work

```
PUT /api/work/{id}
PATCH /api/work/{id}
```

- Parameters:
  - all the work fields described above.

Updates a work. Use *PUT* when updating all the details of a work. Use *PATCH* when updating only some fields.

Delete a Work

```
DELETE /api/works/{id}
```

Marks the work as deleted. The document can be recovered from the Django Admin area, but will never show up in any API otherwise.

Works with linked documents cannot be deleted.

Create a Work

```
POST /api/works
```

- Parameters:
  - all the document fields described above.

1.6.4 Documents

List Documents

```
GET /api/documents
```

Lists the documents in the library. The results will be *paginated*.

Get a Document

```
GET /api/documents/{id}
```

Fetches a JSON description of a document. This does not include the full content or body of the document since those may be very large.

Update a Document

```
PUT /api/documents/{id}
PATCH /api/documents/{id}
```

- Parameters:
  - all the document fields described in *General API Guidelines*

  - `content`: an (optional) content field with the raw XML of the content of the document. `string`

Updates a document. Use *PUT* when updating all the details of a document. Use *PATCH* when updating only some fields.

If you include the `content` parameter, the content of the entire document will be overwritten. Most other fields of the document, such as the FRBR URI and the title will be re-read from the new XML, overwriting any existing fields. The new XML must be valid Akoma Ntoso 2.0 XML.

You can also update the content of the document using `PUT /api/documents/{id}/content`.

Delete a Document

```
DELETE /api/documents/{id}
```

Marks the document as deleted. The document can be recovered from the Django Admin area, but will never show up in any API otherwise.

Create a Document

```
POST /api/documents
```

- Parameters:
  - all the document fields described in *General API Guidelines*
  - `content`: an (optional) content field with the raw XML of the content of the document. `string`
  - `file`: an HTTP file attachment (optional). If this is provided, the content of the document is determined from this file.
  - `file_options..section_number_position`: section number position when `file` is given. One of `before-title`, `after-title` or `guess` (default). Optional. `string`
  - `file_options..cropbox`: crop box for PDF files, as a comma-separated list of integers: `left, top, width, height`. Optional. `string`

The `file` and `file_options` parameters are generally only used when creating a new document. The content of the file will be extracted and parsed. For PDFs, specify a `cropbox` to limit content to within the box on each page.

Use *PUT* when updating all the details of a document. Use *PATCH* when updating only some fields.

Get Document Content

```
GET /api/documents/{id}/content
```

Fetches a JSON description of the raw XML content of a document.

Update Document Content

```
POST /api/documents/{id}/content
```

- Parameters:
  - `content`: raw XML of the document content. `string`

Updates the content of the entire document. Most other fields of the document, such as the FRBR URI and the title will be re-read from the new XML, overwriting any existing fields. The new XML must be valid Akoma Ntoso 2.0 XML.

Warning: This overwrites the entire document. Be careful.

- Parameters:
  - `body`: raw XML of the document body. `string`

Updates the body of the document. The new XML must be valid Akoma Ntoso 2.0 XML `<body>` element.

1.6.5 Attachments

You can attach arbitrary binary files to documents. Each file has a `filename` and `mime_type`.

Note: Attachments are also made available when embedding images into a document. An attachment with a `filename` of `logo.png` is available at `<document url>/media/logo.png`.

List Attachments

```
GET /api/documents/{id}/attachments
```

Fetches a JSON description of the attachments to a document.

Field	Description	Type
id	Unique id of this attachment. Read-only.	Integer
created_at	Timestamp of when the attachment was first created. Read-only.	ISO8601
download_url	URL for downloading this attachment. Read-only.	URL
filenmae	Name for this attachment.	String
mime_type	The mime type of the attachment.	String
size	Size of the attachment in bytes. Read-only.	Integer
updated_at	Timestamp of when the attachment was last updated. Read-only.	ISO8601
url	URL for fetching details of this attachment. Read-only.	URL
view_url	URL for viewing the attachment in-line in the browser, if possible. Read-only.	URL

Create an Attachment

```
POST /api/documents/{id}/attachments
```

Creates a new attachment. Include a `file` multi-part field to upload the binary content.

Update an Attachment

```
PUT /api/documents/{id}/attachments/{id}
PATCH /api/documents/{id}/attachments/{id}
```

Update an attachment.

Delete an Attachment

```
DELETE /api/documents/{id}/attachments/{id}
```

Deletes an attachment.

1.6.6 Helpers

Parse Text into Akoma Ntoso

```
POST /api/parse
```

- Parameters:
  - `file`: an HTTP file attachment (optional). If this is provided, remaining input parameters are ignored. `file`
  - `content`: content to convert. `string`
  - `fragment`: if this is a fragment, not a whole document, the name of the fragment type. Optional. `string`
  - `id_prefix`: prefix to use when generating IDs, especially when parsing a fragment. Optinal. `string`

Parse plain text into Akoma Ntoso. If a `file` is given, then `content` is ignored. Otherwise, `content` is the text to parse. If the content is only a fragment of text, not a full document, then specify the fragment type, such as `sections`, and the prefix to use when generating IDs.

Render Akoma Ntoso into HTML

```
POST /api/render
```

- Parameters:
  - `document`: a document object, included the `content` attribute.

Renders a full document into HTML.

Find and Link Defined Terms

```
POST /api/analysis/link-terms
```

- Parameters:
  - `document`: a document description, only the `content` element is required

Finds defined terms in a document, and finds references to those terms.

Find and Link Referenced Acts

```
POST /api/analysis/link-references
```

- Parameters:
  - `document`: a document description, including the `content` element

Finds and links references to other acts in the document.

1.7 Changelog

1.7.1 3.0 (?)

- FEATURE: support images in documents
- FEATURE: download as XML
- FEATURE: annotations/comments on documents
- FEATURE: download documents as ZIP archives
- You can now highlight lines of text in the editor and transform them into a table, using the Edit > Insert Table menu item.
- Edit menu with Find, Replace, Insert Table, Insert Image, etc.
- Presence indicators for other users editing the same document
- The LIME editor has been removed

1.7.2 2.0 (6 April 2017)

- Upgraded to Django 1.10
- Upgraded a number of dependencies to support Django 1.10
- FEATURE: significantly improved mechanism for maintaining amended versions of documents
- FEATURE: you can now edit tables directly inline in a document
- FEATURE: quickly edit a document section without having to open it via the TOC
- FEATURE: support for newlines in tables
- FEATURE: improved document page layout
- FEATURE: pre-loaded set of publication names per country
- Assent and commencement notices are no longer H3 elements, so PDFs don't include them in their TOCs. #28
- FIX: bug when saving an edited section
- FIX: ensure TOC urls use expression dates
- FIX: faster document saving

After upgrading to this version, you **must** run migrations:

```
python manage.py migrate
```

We also recommend updating the list of countries:

```
python manage.py update_countries_plus
```

1.7.3 1.1 (2016-12-19)

- First tagged release

CHAPTER 2

Documents

A **document** is the primary data type in the platform. It represents a single version of a legislation document such as an act. It contains metadata such as a title, year of publication, country as well as content in Akoma Ntoso XML.

The Akoma Ntoso standard uses an FRBR URI to uniquely identify an item of legislation. For example, the URI

```
/za/act/1998/84
```

identifies the National Forests Act 84 of 1998. The components indicate that the document is from South Africa (`za`), is an `act` and is act number `84` to be published in `1998`. These URIs have a standard format and are well-known and so can be guessed. This is useful when an external system wants to ask us about an act.

See also:

For more details on FRBR URIs, see http://www.akomantoso.org/docs/akoma-ntoso-user-documentation/metadata-describes-the-content

Every Indigo document has an FRBR URI associated with it which is used to identify it in the public REST API.

Every Indigo document also has a uinque numeric identifier `id`. This is assigned by the platform and is unique to that document.

A new document is created each time a new **amendment** must be applied to an existing document. A single piece of legislation that has been amended over time is comprised of multiple separate documents, one for each amendment. Each amended version of the document has a different `id`, but the same URI.

The **application API** uses the numeric IDs to identify documents so that it can handle deleted documents, draft documents and amended documents with the same URI.

The **public API** uses the FRBR URI to identify documents so that anyone can find a document just by constructing a valid FRBR URI.