YANG Cheat Sheet

 Version 1.1 www.yumaworks.com

module-stmt

```
module module-name {
  yang-version 1 | 1.1;
  namespace "uri-string";
  prefix "prefix-string";

  // 0 or more imports
  import "module-name" {
    prefix "prefix-string";
    revision-date "date-string";
    description "revision description"; +
    reference "revision reference"; +
  }

  // 0 or more includes
  include "submodule-name" {
    revision-date "date-string";
    description "revision description"; +
    reference "revision reference"; +
  }

  organization "org-name";
  contact "contact info";
  description "module description";
  reference "module reference";

  // 0 or more revision identifiers
  revision "date-string" {
    description "revision description";
    reference "revision reference";
  }

  body-stmt;   // 0 or more
}
```

body-stmt

```
extension extension-name {
  argument argument-name {
    yin-element true|false;
  }
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
}

feature feature-spec{
  if-feature "feature-spec";  // 0 or more
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
}

identity identity-name {
  base identity-base;  // 0 or 1, 0 or more +
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
}

notification notification-name {
  if-feature "feature-spec"; // 0 or more
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  must-stmt; // 0 or 1 +
  typedef-stmt; // 0 or more
  grouping-stmt; // 0 or more
  data-def-stmt; // 0 or more
}
```

```
grouping grouping-name {
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  typedef-stmt; // 0 or more
  grouping-stmt; // 0 or more
  data-def-stmt; // 0 or more
  action-stmt; // 0 or more +
  notification-stmt; // 0 or more +
}
```

[data-def-stmt]

```
augment "/path/to/augment/target" {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  data-def-stmt | case-stmt | // 1 or more
  action-stmt | notification-stmt; +
}

typedef type-name {
  type-stmt;
  units "units name";
  default "default value";
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
}
```

```
rpc rpc-name {
  if-feature "feature-spec"; // 0 or more
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  typedef-stmt; // 0 or more
  grouping-stmt; // 0 or more
  input {
    must-stmt; // 0 or 1 +
    typedef-stmt; // 0 or more
    grouping-stmt; // 0 or more
    data-def-stmt; // 1 or more
  }
  output {
    must-stmt; // 0 or 1 +
    typedef-stmt; // 0 or more
    grouping-stmt; // 0 or more
    data-def-stmt; // 1 or more
  }
}

deviation "/path/to/deviation/target" {
  description "description text";
  reference "reference text";
  deviate not-supported; // option 1
  deviate operation {  // option 2, 1 or more
    units "units name";
    must "xpath-expr";  // 0 or more
    unique "unique-leafs";  // 0 or more
    default "default value"; // 0 or more +
    type-stmt;
  }
}
```

data-def-stmt

```
container name {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  must "xpath-expr";  // 0 or more
  presence "presence description";
  config true|false;
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  typedef-stmt; // 0 or more
  grouping-stmt; // 0 or more
  data-def-stmt; // 0 or more
  action-stmt; // 0 or more +
  notification-stmt; // 0 or more +
}

leaf name {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  type-stmt;
  units "units name";
  must "xpath-expr";  // 0 or more
  default "default value";
  config true|false;
  mandatory true|false;  // no default if true
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
}

leaf-list name {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  type-stmt;
  units "units name";
  must "xpath-expr";  // 0 or more
  default "default value";  // 0 or more +
  config true|false;
  min-elements number;  // default 0
  max-elements number | unbounded;
  ordered-by user | system;
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
}

case name {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  data-def-stmt; // 0 or more
}
```

```
list name {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  must "xpath-expr";  // 0 or more
  key "key leafs";  // req. if config=true
  unique "unique-leafs";  // 0 or more
  config true|false;
  min-elements number;  // default 0
  max-elements number | unbounded;
  ordered-by user | system;
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  typedef-stmt; // 0 or more
  grouping-stmt; // 0 or more
  data-def-stmt; // 0 or more
  action-stmt; // 0 or more +
  notification-stmt; // 0 or more +
}

choice name {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  default "case name";
  config true|false;
  mandatory true|false;
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  case-stmt | short-case-stmt; // 0 or more
}

short-case-stmt = 1 of
  choice-stmt +
  container-stmt
  leaf-stmt
  leaf-list-stmt
  list-stmt
  anydata-stmt +
  anyxml-stmt

anyxml name {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  must "xpath-expr";  // 0 or more
  config true|false;
  mandatory true|false;
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
}
```

```
anydata name { +
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  must "xpath-expr";  // 0 or more
  config true|false;
  mandatory true|false;
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
}

uses grouping-name {
  when "xpath-expr";
  if-feature "feature-spec"; // 0 or more
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  refine-stmt;  // 0 or more
  uses-augment-stmt;  // 0 or more
}

// not a data-def-stmt
action action-name { +
  if-feature "feature-spec"; // 0 or more
  status current|deprecated|obsolete;
  description "description text";
  reference "reference text";
  typedef-stmt; // 0 or more
  grouping-stmt; // 0 or more
  input {
    must-stmt; // 0 or 1
    typedef-stmt; // 0 or more
    grouping-stmt; // 0 or more
    data-def-stmt; // 1 or more
  }
  output {
    must-stmt; // 0 or 1
    typedef-stmt; // 0 or more
    grouping-stmt; // 0 or more
    data-def-stmt; // 1 or more
  }
}
```

Key

blue text = keyword
green text = string content
red text = comment
italics = optional
+ = YANG version 1.1

type-stmt

```
type integer{
  range "range-expr";  // 0 or 1
}

// integer = [u]int (8, 16, 32, 64)

type decimal64 {
  fraction-digits "integer";  // value is 1..18
  range "range-expr";  // 0 or 1
}

type string {
  length "range-expr";  // 0 or 1
  pattern "pattern-expr";  // 0 or more
}

type enumeration {
  enum "enum-string"  {
    if-feature "feature-spec";  // 0 or more +
    value integer;
    status current|deprecated|obsolete;
    description "description text";
    reference "reference text";
  } // 1 or more
}

type bits {
  bit "bit-name"  {
    if-feature "feature-spec";  // 0 or more +
    position "integer";
    status current|deprecated|obsolete;
    description "description text";
    reference "reference text";
  }  // 1 or more
}

type identityref {
  base "identity-base";  // 1, 1 or more +
}

type leafref {
  path "path-expr";
  require-instance true|false; +
}

type boolean;
type empty;
type binary {
  length "range-expr";  // 0 or 1
}

type union {
  type-stmt;  // 1 or more
}

type instance-identifier {
  require-instance true|false:
}
```