{{## def.setupKeyword:

{{

var $lvl = it.level;

var $dataLvl = it.dataLevel;

var $schema = it.schema[$keyword];

var $schemaPath = it.schemaPath + it.util.getProperty($keyword);

var $errSchemaPath = it.errSchemaPath + '/' + $keyword;

var $breakOnError = !it.opts.allErrors;

var $errorKeyword;

var $data = 'data' + ($dataLvl || '');

var $valid = 'valid' + $lvl;

var $errs = 'errs\_\_' + $lvl;

}}

#}}

{{## def.setCompositeRule:

{{

var $wasComposite = it.compositeRule;

it.compositeRule = $it.compositeRule = true;

}}

#}}

{{## def.resetCompositeRule:

{{ it.compositeRule = $it.compositeRule = $wasComposite; }}

#}}

{{## def.setupNextLevel:

{{

var $it = it.util.copy(it);

var $closingBraces = '';

$it.level++;

var $nextValid = 'valid' + $it.level;

}}

#}}

{{## def.ifValid:

{{? $breakOnError }}

if ({{=$valid}}) {

{{ $closingBraces += '}'; }}

{{?}}

#}}

{{## def.ifResultValid:

{{? $breakOnError }}

if ({{=$nextValid}}) {

{{ $closingBraces += '}'; }}

{{?}}

#}}

{{## def.elseIfValid:

{{? $breakOnError }}

{{ $closingBraces += '}'; }}

else {

{{?}}

#}}

{{## def.nonEmptySchema:\_schema:

(it.opts.strictKeywords

? (typeof \_schema == 'object' && Object.keys(\_schema).length > 0)

|| \_schema === false

: it.util.schemaHasRules(\_schema, it.RULES.all))

#}}

{{## def.strLength:

{{? it.opts.unicode === false }}

{{=$data}}.length

{{??}}

ucs2length({{=$data}})

{{?}}

#}}

{{## def.willOptimize:

it.util.varOccurences($code, $nextData) < 2

#}}

{{## def.generateSubschemaCode:

{{

var $code = it.validate($it);

$it.baseId = $currentBaseId;

}}

#}}

{{## def.insertSubschemaCode:

{{= it.validate($it) }}

{{ $it.baseId = $currentBaseId; }}

#}}

{{## def.\_optimizeValidate:

it.util.varReplace($code, $nextData, $passData)

#}}

{{## def.optimizeValidate:

{{? {{# def.willOptimize}} }}

{{= {{# def.\_optimizeValidate }} }}

{{??}}

var {{=$nextData}} = {{=$passData}};

{{= $code }}

{{?}}

#}}

{{## def.$data:

{{

var $isData = it.opts.$data && $schema && $schema.$data

, $schemaValue;

}}

{{? $isData }}

var schema{{=$lvl}} = {{= it.util.getData($schema.$data, $dataLvl, it.dataPathArr) }};

{{ $schemaValue = 'schema' + $lvl; }}

{{??}}

{{ $schemaValue = $schema; }}

{{?}}

#}}

{{## def.$dataNotType:\_type:

{{?$isData}} ({{=$schemaValue}} !== undefined && typeof {{=$schemaValue}} != \_type) || {{?}}

#}}

{{## def.check$dataIsArray:

if (schema{{=$lvl}} === undefined) {{=$valid}} = true;

else if (!Array.isArray(schema{{=$lvl}})) {{=$valid}} = false;

else {

#}}

{{## def.numberKeyword:

{{? !($isData || typeof $schema == 'number') }}

{{ throw new Error($keyword + ' must be number'); }}

{{?}}

#}}

{{## def.beginDefOut:

{{

var $$outStack = $$outStack || [];

$$outStack.push(out);

out = '';

}}

#}}

{{## def.storeDefOut:\_variable:

{{

var \_variable = out;

out = $$outStack.pop();

}}

#}}

{{## def.dataPath:(dataPath || ''){{? it.errorPath != '""'}} + {{= it.errorPath }}{{?}}#}}

{{## def.setParentData:

{{

var $parentData = $dataLvl ? 'data' + (($dataLvl-1)||'') : 'parentData'

, $parentDataProperty = $dataLvl ? it.dataPathArr[$dataLvl] : 'parentDataProperty';

}}

#}}

{{## def.passParentData:

{{# def.setParentData }}

, {{= $parentData }}

, {{= $parentDataProperty }}

#}}

{{## def.iterateProperties:

{{? $ownProperties }}

{{=$dataProperties}} = {{=$dataProperties}} || Object.keys({{=$data}});

for (var {{=$idx}}=0; {{=$idx}}<{{=$dataProperties}}.length; {{=$idx}}++) {

var {{=$key}} = {{=$dataProperties}}[{{=$idx}}];

{{??}}

for (var {{=$key}} in {{=$data}}) {

{{?}}

#}}

{{## def.noPropertyInData:

{{=$useData}} === undefined

{{? $ownProperties }}

|| !{{# def.isOwnProperty }}

{{?}}

#}}

{{## def.isOwnProperty:

Object.prototype.hasOwnProperty.call({{=$data}}, '{{=it.util.escapeQuotes($propertyKey)}}')

#}}