

L'extension *intexgral**

Valentin Dao[†]

Publiée le 26-07-2025

Résumé

Tandis que la composition d'intégrales est très fréquente en \LaTeX , cette dernière s'avère souvent peu pratique. Lorsque l'expression se complexifie, il devient alors difficile d'en modifier les différents éléments dans un code source peu lisible. Pour répondre à ce problème, le package *intexgral* met à disposition une macro centrale dont le seul argument est l'intégrande. Tout le reste (les symboles, les bornes, les variables d'intégration...) pourra aisément être modifié grâce à une interface $\langle \text{clé}=\text{valeur} \rangle$. Contrairement à la méthode classique, où l'utilisateur choisit l'ordre des bornes avec les caractères actifs $_$ et $^$, le package a dû fixer une convention. Ainsi, pour les deux clés gérant les bornes qui seront présentées, l'entrée supposée sera $_{\langle \text{borne inférieure} \rangle}^{\langle \text{borne supérieure} \rangle}$. Ce package étant écrit en *expl3*, il dépend donc des packages \LaTeX3 *l3kernel* et *l3package*. De plus, *intexgral* utilise le package *derivative*¹ afin de faciliter la manipulation de différentielles, ainsi que le package *pkginfograb*².

*Ce fichier décrit la version v3.0.0, mise à jour le 24-12-2025

[†]E-mail : vdao.texdev@gmail.com

1. Disponible sur CTAN sous : <https://ctan.org/pkg/derivative>

2. Disponible sur CTAN sous : <https://ctan.org/pkg/pkginfograb>

Table des matières

Résumé	1
1 Options de package	4
2 Présentation de la macro principale	4
3 Liste des clés	5
3.1 Bornes d'intégration	5
3.1.1 Définir les bornes	5
3.2 Modes d'affichage	6
3.3 Symbole (et encore des bornes)	7
3.3.1 Sélectionner le symbole	7
3.3.2 Générer beaucoup d'intégrales	7
3.3.3 Définir des bornes ponctuellement	8
3.3.4 Motifs récurrents de bornes	8
3.4 Différentielles	9
3.4.1 Spécifier les variables d'intégration	9
3.4.2 Inclure le jacobien	9
3.4.3 Modifier le symbole des différentielles	10
3.4.4 Réaliser une intégrale curviligne	10
3.4.5 Opter pour la variante étoilée	10
3.4.6 Désigner des options	10
4 Macros annexes	11
4.1 Emplacements personnalisés des différentielles	11
4.2 Retour sur la clé <i>limits</i>	12
4.3 Retour sur la clé <i>variables</i>	13
4.4 Retour sur la clé <i>symbol</i>	14
5 Syntaxe <i>spéciale</i>	15
5.1 Présentation de la syntaxe	15
5.2 Fonctionnement de la syntaxe	16
6 Paramètres annexes	16
Historique des modifications	18
Index	20
Implémentation	26

Licence

© 2025 Valentin Dao, publié sous la \LaTeX Project Public License (LPPL) 1.3c

Polices

Chronicle Text G3 Roman

© 2002, 2007 Hoefler & Frere-Jones.

Whitney-Medium

© 1996, 2009 Hoefler & Frere-Jones.

Monospace Argon Medium

© 2023, GitHub <https://github.com/githubnext/monospace>

Dépôt

 [Voir dépôt GitHub](https://github.com).

Remerciements

Un grand merci à Plante et Slurpy de m'avoir accompagné dans cette aventure que fut l'apprentissage de \TeX , vos conseils ont été précieux. Je tiens également à remercier Anthony pour avoir toujours gentiment révisé les nouvelles versions et m'avoir donné des idées pour celles à venir.

1 Options de package

`\intexgralsetup` `\intexgralsetup` {<options de package>}

nouveau: v2.0.0

Ces options peuvent être déclarées de façon classique avec `\usepackage` ou bien avec la macro ici présente dans le préambule :

`invert-limits` `invert-limits=<true|false>`

Cette clé permet d'inverser la convention d'ordre des limites. Le document ne peut suivre qu'une seule convention.

`invert-diff` `invert-diff=<true|false>`

mise à jour: v3.0.0

Il est courant de voir les variables d'intégration placées avant l'intégrande dans les articles de physique. Cette clé intervertit donc leurs positionnements.

`limits-mode` `limits-mode=<limits|nolimits>`

nouveau: v3.0.0

Applique `\limits` ou `\nolimits` à toutes les intégrales.

`italic` `italic=<true|false>`

`upright` `upright=<true|false>`

Ce sont les deux clés du package *derivative*.

2 Présentation de la macro principale

`\integral` `\integral` [<list de clés>] {<intégrande>}

mise à jour: v3.0.0

Cette macro est celle qui permet de composer les intégrales. Elle doit naturellement être utilisée en mode mathématique uniquement. En voici un premier exemple sous sa forme la plus simple :

```
\begin{equation}
  \integral{x}
\end{equation}
```

$$\int x \, dx \tag{1}$$

Puisque cette macro est axée autour de l'utilisation de clés, la première partie de cette documentation les présentera en les regroupant par domaines. La seconde partie quant à elle introduira les quelques macros annexes qui viennent compléter l'usage de certaines clés. Le reste de ce document exposera les autres fonctionnalités du package.

3 Liste des clés

3.1 Bornes d'intégration

3.1.1 Définir les bornes

`limits` `limits=` {<liste-mixte>}, <mot-clé>
`limits*` `limits*=` {<liste-mixte>}, <mot-clé>

Cette clé détermine les bornes d'intégration à utiliser en suivant la convention édictée dans le résumé. Sous sa forme la plus simple, la valeur de la clé prend comme argument une liste d'éléments séparés par une virgule (*comma-separated values* ou <csv-list>¹).

```
\begin{equation}
  \integral[limits={1, 10}]{f(x)}
\end{equation}
```

$$\int_1^{10} f(x) \, dx \quad (2)$$

L'avantage notable de la clé `limits` est qu'on peut lui spécifier les bornes d'intégration de plusieurs intégrales, et le package compose automatiquement les symboles correspondants. Pour séparer les paires de bornes, il faut utiliser le point-virgule (*semicolon-separated values* ou <ssv-list>¹).

```
\begin{equation}
  \integral[limits={1, 2; 3, 4}, variables={x, y}]{f(x, y)}
\end{equation}
```

$$\int_1^2 \int_3^4 f(x, y) \, dx \, dy \quad (3)$$

Il est également possible de renseigner des bornes prédéfinies à l'aide de mots-clés (voir section 4.2). Enfin, la variante étoilée permet d'exprimer les bornes de l'intégrale sous forme d'intervalle. Le cas échéant, le sens des crochets s'adapte automatiquement à la présence de $\pm\infty$

```
\begin{equation}
  \integral[limits*={1, 10}]{f(x)}
\end{equation}
```

$$\int_{[1,10]} f(x) \, dx \quad (4)$$

1. On entend donc par <liste-mixte> le fait que `limits` puisse accepter un ensemble de <csv-list> dans une <ssv-list> plus globale.

3.2 Modes d’affichage

mode `mode=\langle\text{default}|\text{nested}|\text{product}\rangle`

nouveau: v3.0.0 *Uniquement* lorsque la clé `limits` est employée, il est possible d’alterner entre trois styles d’affichage distincts. Chacun d’entre eux est entièrement compatible avec l’option `invert-diff`.

default Compose l’ensemble des symboles, puis l’intégrande, puis les variables d’intégration. La macro opérant par défaut dans ce mode, il n’est donc pas nécessaire d’utiliser la clé avec cette valeur.

nested Compose une intégrale *imbriquée* où symbole et intégrande alternent avant que tous les variables d’intégrations soient écrits.

product Compose un produit d’intégrales où symbole, intégrande et variables d’intégration alternent.

Bien sûr, rien ne vous empêche pour le mode produit d’utiliser plusieurs fois la macro `\integral` d’affilée pour produire le même résultat que dans l’exemple ci-dessous. Pour celles et ceux qui préféreraient cependant obtenir le résultat avec une seule macro, cette méthode est permise.

important: Pour les deux derniers modes, l’intégrande sera, d’une certaine manière, *découpée*. Afin de correctement effectuer cette action, il faut avoir recours à la même méthode qu’avec `limits`, c’est-à-dire en employant le point-virgule.

MODE **default**

```
\begin{equation}
  \integral[limits={1, 2; 3, 4; 5, 6}, variables={x, y, z}, mode=default]{xyz}
\end{equation}
```

$$\int_1^2 \int_3^4 \int_5^6 xyz \, dx \, dy \, dz \quad (5)$$

MODE **nested**

```
\begin{equation}
  \integral[limits={1, 2; 3, 4; 5, 6}, variables={z, y, x}, mode=nested]{x;y;z}
\end{equation}
```

$$\int_1^2 x \int_3^4 y \int_5^6 z \, dz \, dy \, dx \quad (6)$$

MODE **product**

```
\begin{equation}
  \integral[limits={1, 2; 3, 4; 5, 6}, variables={x, y, z}, mode=product]{x;y;z}
\end{equation}
```

$$\int_1^2 x \, dx \int_3^4 y \, dy \int_5^6 z \, dz \quad (7)$$

Pour bien fonctionner, il est évident que les clés `limits` et `variables` (voir 3.4.1), ainsi que l'intégrande, doivent avoir le même nombre d'éléments. Si ce n'est pas le cas, la compilation n'échouera pas, mais l'expression de l'intégrale risque de ne plus avoir aucun sens. De nombreux messages d'avertissement seront là pour vous en informer.

3.3 Symbole (et encore des bornes)

Les clés `limits(*)` se révèlent très utiles lorsque l'on souhaite composer une intégrale définie. Néanmoins, cela couvre seulement les expressions finales où les bornes de chaque variable ont été spécifiées. Pour des cas plus généraux — les intégrales indéfinies donc — elles sont relativement malcommodes. Pour composer une intégrale double sur une surface S , on devrait écrire `limits={, ; S, }`. Il va sans dire que c'est assez mal adapté pour l'utilisateur et peu optimal pour le package. Par ailleurs, le glyphe ne serait pas correct. L'ensemble des clés à suivre propose donc une façon de facilement modifier le symbole ainsi que les bornes.

3.3.1 Sélectionner le symbole

`symbol`

`symbol=<séquence de contrôle>`

mise à jour: v3.0.0

Cette clé accepte une macro désignant un symbole d'intégration. Tout symbole préalablement défini par une séquence de contrôle est recevable. Si vous tentez d'en utiliser une qui n'est pas définie, elle sera substituée à `\iint` et un message d'avertissement sera émis.

remarque : À part la vérification de l'existence de la macro, aucun contrôle particulier n'est effectué et la valeur de la clé est utilisée telle quelle pour composer le symbole. Ceci implique notamment deux choses. Tout d'abord, le symbole va dépendre de la façon dont il est défini. Par exemple, `amsmath` et `unicode-math` ne définissent pas `\iint` de la même manière. Le résultat sera donc différent. De plus, si vous renseignez comme argument de la clé une macro *définie* mais ne correspondant pas à un symbole intégral, le package n'en tiendra pas compte et utilisera quand même le symbole donné. Outre l'incohérence mathématique que cela peut représenter, il est susceptible que cela mène, sous certaines circonstances, à des erreurs de bas niveau.

```
\begin{equation}
\integral[symbol=\iint, llimit=S, variables={x, y}]{f(x, y)}
\end{equation}
```

$$\iint_S f(x, y) \, dx \, dy \quad (8)$$

3.3.2 Générer beaucoup d'intégrales

`nint` `nint=<entier>`

Cette clé accepte un entier n qui permet de composer n intégrales. Il est conseillé d'avoir recours à cette clé seulement si le nombre de symboles dépasse 4 afin de privilégier les glyphes définis par la police mathématique utilisée.

```
\begin{equation}
\int\limits[nint=5, llimit=\Omega, variables={x_1, x_2, x_3, x_4, x_5}]{f(x_1, x_2,
x_3, x_4, x_5)}
\end{equation}
```

$$\iiint\limits_{\Omega} f(x_1, x_2, x_3, x_4, x_5) dx_1 dx_2 dx_3 dx_4 dx_5 \quad (9)$$

3.3.3 Définir des bornes ponctuellement

```
llimit    llimit=<borne inférieure>
ulimit    ulimit=<borne supérieure>
```

mise à jour: v3.0.0 Ces deux clés permettent de spécifier les bornes inférieures et supérieures respectivement. Elles ne sont adaptées que si un seul symbole est affiché. Si vous avez besoin de spécifier les deux bornes, la clé `limits` devra être privilégiée.

```
\begin{equation}
\int\limits[llimit={x^2 + y^2 \leq 1}, variables={x, y}]{f(x, y)}
\end{equation}
```

$$\int_{x^2+y^2 \leq 1} f(x, y) dx dy \quad (10)$$

3.3.4 Motifs récurrents de bornes

Les bornes suivent parfois des schémas courants qui peuvent se généraliser avec des clés, évitant ainsi de surcharger l'argument de `llimit`.

```
domain    domain= {<*-liste>}
domain*   domain*= {<*-liste>}
```

mise à jour: v3.0.0 Ces clés acceptent une liste dont le délimiteur est un astérisque. Ensuite, chaque élément de la liste est analysé de la façon suivante :

- Le premier token de l'item se voit passer `\mathbb` (précédé d'`\uppercase` au cas où la touche SHIFT serait trop loin pour vos doigts).
- Les tokens restants — qui peuvent être vides — sont placés comme exposant (ou en indice pour la variante étoilée de la clé).

```
\begin{equation}
\int\limits[symbol=\iint, domain={\mathbb R*\mathbb R}, variables={x, y}]{xy}
\end{equation}
```

$$\iint_{\mathbb R \times \mathbb R} xy dx dy \quad (11)$$

```
boundary boundary= {<borne inférieure>}
```

Cette clé place simplement le symbole ∂ avant la borne inférieure.

```
\begin{equation}
  \integral[symbol=\oint, boundary=S, diff-vec]{G(\vec r)}
\end{equation}
```

$$\oint_{\partial S} G(\vec{r}) \cdot d\vec{r} \quad (12)$$

3.4 Différentielles

3.4.1 Spécifier les variables d'intégration

variables `variables={⟨liste-csv⟩, ⟨mot-clé⟩, none`

mise à jour: v3.0.0 Cette clé permet de définir les variables d'une intégrale sous forme d'une `⟨csv-list⟩`. La clé peut, tout comme `limits`, accepter un mot-clé comme argument. Ce comportement est aussi expliqué plus tard (voir section 4.3)

remarque : Si aucune variable d'intégration n'est renseignée, c'est-à-dire que `variables` n'est pas appelée, le package place automatiquement « dx » (ou « $d\vec{r}$ » si `diff-vec` est actif). De plus, si `none` est passé comme valeur, aucune variable d'intégration ne sera affichée.

```
\begin{equation}
  \integral[variables=t]{t^2}
\end{equation}
```

$$\int t^2 dt \quad (13)$$

3.4.2 Inclure le jacobien

jacobian `jacobian`

Cette clé active l'affichage du jacobien lorsque défini par `\NewVariableKeyword`.

```
\begin{equation}
  \integral[limits={0, R; 0, 2\pi; 0, \pi}, variables=spherical, mode=product,
  jacobian]{;}
\end{equation}
```

$$\int_0^R r^2 dr \int_0^{2\pi} \sin \theta d\theta \int_0^\pi d\phi \quad (14)$$

Comme évoqué plus tôt, `derivative` est utilisé par le package pour composer les différentielles. L'ensemble des clés à suivre, dont le nom portera le préfixe «`diff-`»¹, vous permettent ainsi d'appliquer les fonctionnalités de la macro `\odiff`.

1. Quoique la clé `diff-vec` ne soit pas liée à `derivative`.

3.4.3 Modifier le symbole des différentielles

diff-symb diff-symb= \langle macro de différentielle \rangle

Cette clé permet de modifier le style des différentielles parmi ceux définis par `\NewDifferential`.

```
\NewDifferential{\feynmandiff}{\mathcal{D}}
\begin{equation}
\integral[variables={q(t)}, diff-symb=\feynmandiff]{e^{+\dfrac{\imath}{\hbar} S[q(t)]}}{
\hbar}
\end{equation}
```

$$\int e^{+\frac{iS[q(t)]}{\hbar}} \mathcal{D}q(t) \quad (15)$$

3.4.4 Réaliser une intégrale curviligne

diff-vec diff-vec

Cette clé applique un vecteur à chaque variable d'intégration en plus d'un point médian. Elle n'est compatible qu'avec le mode `default`.

```
\begin{equation}
\int[double=S, variables=S, diff-vec]{\vec F}
\end{equation}
```

$$\iint_S \vec{F} \cdot d\vec{S} \quad (16)$$

3.4.5 Opter pour la variante étoilée

diff-star diff-star

Activer cette clé revient à utiliser `\odif*` ou ses variantes.

```
\begin{equation}
  \integral[double, variables={x, y}, diff-star]{x^2 \exp(y)}
\end{equation}
```

$$\iint x^2 \exp(y) \, \mathrm{d}_x \mathrm{d}_y \quad (17)$$

3.4.6 Désigner des options

diff-options diff-options= {<liste clé-valeur>}

Cette clé accepte une liste de clés telle qu'elle aurait été écrite comme argument optionnel de `\odif` ou ses variantes. Pour illustrer, `diff-options={order={2, 3}, var=all}` sera interprété comme `\odif[order={2, 3}, var=all]{...}`.

```
\begin{equation}
\integral[diff-options={order=4}]{\bar{\psi}(x)(\imath\gamma^\mu\partial_\mu-m)\psi(
x)}
\end{equation}
```

$$\int \bar{\psi}(x)(\imath\gamma^\mu\partial_\mu - m)\psi(x) d^4x \quad (18)$$

4 Macros annexes

En plus de la large gamme de clés définies par le package, quelques macros viennent également améliorer leurs usages.

4.1 Emplacements personnalisés des différentielles

`\differentials` `\differentials`

Bien que l’option `invert-diff` existe, il est souvent souhaitable de pouvoir placer les différentielles où l’on veut. Par exemple, il est fréquent de les voir au numérateur d’une fraction lorsque celui-ci vaut 1. Pour répondre à ce besoin, le package met à disposition la macro `\differentials`. Pour les modes `default` et `nested`, la macro compose toutes les différentielles d’un coup. Avec `product`, il faudra répéter la macro entre chaque point-virgule.

```
\begin{equation}
\integral{\frac{\differentials}{x}}
\end{equation}
```

$$\int \frac{dx}{x} \quad (19)$$

4.2 Retour sur la clé *limits*

```
\NewLimitsKeyword    \NewLimitsKeyword {\mot-clé} {\bornes}
\RenewLimitsKeyword
\ProvideLimitsKeyword
\DeclareLimitsKeyword
```

Il est courant de devoir renseigner les mêmes bornes dans une intégrale. Pour faciliter leur écriture, les clés `limits(*)` acceptent, en plus des bornes explicites, un mot-clé en désignant une paire prédéfinie par l'une de ces quatre macros :

- `\NewLimitsKeyword` Crée un nouveau mot-clé et émet une erreur s'il existe déjà.
- `\RenewLimitsKeyword` Redéfinit un mot-clé et émet une erreur s'il n'existe pas déjà.
- `\ProvideLimitsKeyword` Ne crée un nouveau mot-clé que s'il n'existe pas déjà. Aucun message ne sera émis dans le cas échéant.
- `\DeclareLimitsKeyword` Crée un nouveau mot-clé quoi qu'il en soit, écrasant toute définition préalable.

Voici la liste des mots-clés déjà définis par le package et les bornes qu'ils contiennent :

<code>ab</code>	a, b
<code>unit</code>	$0, 1$
<code>real</code>	$-\infty, \infty$
<code>positive</code>	$0, \infty$
<code>negative</code>	$-\infty, 0$
<code>circle</code>	$0, 2\pi$
<code>sircle</code>	$0, \pi$
<code>qcircle</code>	$0, \frac{\pi}{2}$
<code>height</code>	$0, H$
<code>radius</code>	$0, R$
<code>length</code>	$0, L$
<code>time</code>	$0, T$

remarque : Les mots-clés contiennent les *deux* bornes d'une intégrale en même temps. Ils doivent donc être précédés et/ou suivis de point-virgule.

important : Dû à l'implémentation de `\NewLimitsKeyword` et ses variantes, l'utilisateur *doit* suivre la convention d'ordre des bornes du package, même si `invert-limits` est fixé à `true`

On pourrait donc modifier l'expression d'une l'intégrale de la façon suivante :

```
\begin{equation}
  \integral[limits={radius;circle;scircle}, variables={\rho, z, \phi}]{\rho z \phi}
\end{equation}
```

$$\int_0^R \int_0^{2\pi} \int_0^\pi \rho z \phi \, d\rho \, dz \, d\phi \quad (20)$$

Il est tout à fait possible de mélanger ces mots-clés avec la syntaxe plus explicite de `limits`.

```
\begin{equation}
  \int\limits[limits={time; 34, 40; height}, variables={h, \omega, t}]{h \omega t}
\end{equation}
```

$$\int_0^T \int_{34}^{40} \int_0^H h \omega t \, dh \, d\omega \, dt \quad (21)$$

4.3 Retour sur la clé *variables*

```
\NewVariableKeyword      \NewVariableKeyword {<mot-clé>} {<variables>} [<jacobien>]
\RenewVariableKeyword
\ProvideVariableKeyword
\DeclareVariableKeyword
```

De façon similaire, les mêmes groupes de variables réapparaissent souvent. On peut donc également définir des mots-clés contenant un ensemble de variables d'intégration préenregistrés. Les variantes de cette macro ont le même comportement que pour `\NewLimitsKeyword`. La seule différence est qu'il est ici possible de définir un jacobien, sous forme d'une `<csv-list>` si besoin. Son affichage est ensuite contrôlé par la clé `jacobian` expliquée précédemment. Voici la liste des mots-clés de variables déjà définis par le package, avec le jacobien pour certains :

```
cartesian    x, y, z
planar       x, y
polar        r, \theta(r)
cylindrical  r, \theta, z(r)
spherical    r, \theta, \phi(r^2, \sin \theta)
```

```
\begin{equation}
  \int\limits[triple=V, variables=spherical]{f(x, y, z)}
\end{equation}
```

$$\iiint_V f(x, y, z) \, dr \, d\theta \, d\phi \quad (22)$$

4.4 Retour sur la clé *symbol*

```

\NewSymbolKeyword    \NewSymbolKeyword {⟨clé⟩} {⟨symbole⟩}
\RenewSymbolKeyword
\ProvideSymbolKeyword
\DeclareSymbolKeyword

```

nouveau: v3.0.0

Afin de composer des intégrales indéfinies, le package offre essentiellement deux clés : `symbol` et `llimit`. Bien qu'elles soient simples d'utilisation, il demeure toujours laborieux de les écrire toutes les deux avec leurs valeurs. C'est pourquoi *intexgral* met à disposition des clés dites *raccourcies*, dont le but est de combiner l'action de sélection du symbole et des bornes. Contrairement aux deux macros précédentes, il s'agit ici de créer de toutes nouvelles clés, et non simplement des valeurs spécifiques attribuées à `symbol`. Toutes seules, ces clés modifient le symbole intégral. La valeur de ces clés correspondra elle à la borne inférieure. Voici la liste de l'ensemble des *clé-symbole* définie par le package :

<code>single</code>	symbole utilisé = <code>\int</code>
<code>double</code>	symbole utilisé = <code>\iint</code>
<code>triple</code>	symbole utilisé = <code>\iiint</code>
<code>quadruple</code>	symbole utilisé = <code>\iiiiint</code>
<code>contour</code>	symbole utilisé = <code>\oint</code>
<code>surface</code>	symbole utilisé = <code>\oiint</code>
<code>volume</code>	symbole utilisé = <code>\oiint</code>

important : Offrir la possibilité de créer soi-même une clé à la macro `\integral` représente un risque qui sera mieux expliqué dans la prochaine section. Retenez pour l'instant que toutes ces macros¹ émettront un message d'avertissement si vous tentez de créer une nouvelle *clé-symbole* portant le même nom qu'un groupe de bornes définies.

```

\begin{equation}
  \integral[contour=\mathcal{C}, diff-vec]{f(\vec r)}
\end{equation}

```

$$\oint_{\mathcal{C}} f(\vec{r}) \cdot d\vec{r} \quad (23)$$

5 Syntaxe spéciale

5.1 Présentation de la syntaxe

`\integral` `\integral` [`(limits:variables(+j):mode)`] {`(intégrande)`}

nouveau: v3.0.0

En ce qui concerne les intégrales définies, l'utilisateur sera amené à employer au maximum quatre clés pour les composer : `limits`, `variables`, `mode` et `jacobian`. On peut néanmoins leur reprocher la même chose qu'avant ; écrire ces quatre clés, pouvant recevoir des arguments assez longs, va à l'encontre de l'objectif principal de ce package. Ainsi, l'utilisateur peut désigner comme argument optionnel d'`\integral`, une syntaxe dite *spéciale* qui n'est propre qu'à `intexgral`. La logique est la suivante :

- Vous spécifiez un argument *valide* de la clé `limits`
- Vous spécifiez un argument *valide* de la clé `variables`
- Vous spécifiez un argument *valide* de la clé `mode`

Et vous séparez le tout d'un deux points. Voyons un exemple pour mieux comprendre.

```
\begin{equation}
  \integral[1, 2; 4, 5:y, x:nested]{x; y}
\end{equation}
```

$$\int_1^2 x \int_4^5 y \, dy \, dx \quad (24)$$

On entend par *argument valide* le fait que toutes les formes de valeurs acceptées par les quatre clés peuvent être pareillement adoptées par la syntaxe spéciale. Cela comprend donc les mots-clés.

```
\begin{equation}
  \integral[radius;circle;scircle:spherical:product]{r;\theta;\phi}
\end{equation}
```

$$\int_0^R r \, dr \int_0^{2\pi} \theta \, d\theta \int_0^\pi \phi \, d\phi \quad (25)$$

Afin d'inclure le jacobien, il suffit d'écrire `+j` après l'argument de `variables`. Le mode peut aussi être indiqué à l'aide d'une initiale seulement.

```
\begin{equation}
  \integral[radius;circle;scircle:spherical+j:p]{;};
\end{equation}
```

$$\int_0^R r^2 \, dr \int_0^{2\pi} \sin \theta \, d\theta \int_0^\pi d\phi \quad (26)$$

Il est important de répéter que dans la configuration classique de l'argument optionnel, il n'est bien sûr pas obligatoire de renseigner les quatre clés citées. Il en va de même pour cette syntaxe. Puisque la clé `mode` n'est pas nécessaire pour `default`, cette partie peut être omise.

```
\begin{equation}
  \integral[1, 2; 3, 4:x, y]{xy}
\end{equation}
```

$$\int_1^2 \int_3^4 xy \, dx \, dy \quad (27)$$

Vous pouvez aussi toujours profiter du placement automatique du « dx » avec cette syntaxe en faisant abstraction de `variables`.

```
\begin{equation}
  \integral[1, 10]{x}
\end{equation}
```

$$\int_1^{10} x \, dx \quad (28)$$

5.2 Fonctionnement de la syntaxe

Du point de vue de l'utilisateur, il est relativement simple de choisir quelle syntaxe adopter. Toutefois, le package doit pouvoir distinguer les deux. Sans trop rentrer dans les détails¹ de l'implémentation, il est porté à votre intention que le package tente d'extraire la première clé de l'argument optionnel² et vérifie si elle existe. C'est pourquoi, si vous créez une clé avec `\NewSymbolKeyword` dont le nom pourra probablement servir de bornes d'intégrations, *intexrgal* peut faussement évaluer la nature de l'argument optionnel.

6 Paramètres annexes

`\IntegralSetup` `\IntegralSetup` {<liste de paramètres>}

nouveau: v3.0.0

Cette macro permet, sous la syntaxe <clé=valeur>, de modifier des paramètres liés à certaines clés. Toutes les assignations effectuées sont locales et la macro peut donc être placée dans un groupe si besoin. Voici un exemple de la macro avec les assignations par défaut du package :

```
\IntegralSetup
{
  defaultvar = {x},
  defaultvar* = {x},
  vectorstyle = \vec,
  domainstyle = \mathbb,
  symbolskip = {-6mu},
  hide-diff = false
}
```

-
1. Les plus courageux d'entre vous sont tout de même invités à lire le code source.
 2. Ou du moins, le groupe de tokens qu'il pense correspondre à une clé.

defaultvar `defaultvar={⟨variables⟩}`

defaultvar* Cette clé modifie la variable d'intégration insérée automatiquement lorsque `variables` n'est pas utilisée. L'argument peut tout à fait correspondre à un mot clé défini par `\NewVariableKeyword`. La variante étoilée de la clé contrôle la variable d'intégration à composer avec `diff-vec`.

vectorstyle `vectorstyle=⟨séquence de contrôle⟩`

Cette clé modifie la macro à appliquer aux variables quand `diff-vec` est en action. Il est donc possible d'altérer le style du vecteur : gras, souligné, ou même un autre style d'un package particulier — `esvect` par exemple.

domainstyle `domainstyle=⟨séquence de contrôle⟩`

Semblablement, on peut changer la macro à appliquer pour les clés `domain(*)`.

symbolskip `symbolskip=⟨muexpr⟩`

Lorsque le mode `default` est en vigueur, le package insère un crénage entre les symboles pour légèrement les rapprocher. La dimension par défaut (-6μ) peut ainsi être ajustée selon la police mathématique en vigueur.

hide-diff `hide-diff=⟨true|false⟩`

Lorsqu'une large partie du document nécessite de ne pas inclure les différentielles, il est possible de prolonger dans la durée l'action plus ponctuelle de `variables=none`.

Historique des modifications

3.0.0 (09-12-2025)

Ajouté

- ▶ Syntaxe spéciale.
- ▶ Clés `domain*` et `mode`.
- ▶ Macros `\IntegralSetup` et `\NewSymbolKeyword`.

Supprimé

- ▶ Macros `\defaultdiff`, `\defaultdiff`, `\defaultvdiff` et `\vdiffstyle` en faveur de `\IntegralSetup`.
- ▶ Clés contrôlant symbole et borne, maintenant gérées à un plus haut niveau avec `\NewSymbolKeyword`.
- ▶ Clé `int-split`, en faveur de `mode`.
- ▶ Macro `\NewIntegralSymbol`.

Modifié

- ▶ Les noms de quelques clés (`variable` en `variables`, `lower-lim` et `upper-lim` en `llimit` et `ulimit`, `int-symb` en `symbol`, `invert-differentials` et `hide-differentials` en `invert-diff` et `hide-diff`).
- ▶ `jacobian` et `diff-star` ne nécessitent plus de valeur booléenne. Il suffit maintenant de les écrire pour activer leurs fonctionnalités.
- ▶ `hide-diff` attribué à une option locale plutôt qu'à une option de package.
- ▶ `limits-mode` attribué à une option de package plutôt qu'à une clé de macro.

v2.0.1 (13-09-2025)

Corrigé

- ▶ Problème de compatibilité entre `unicode-math` et `amssymb` selon l'ordre de chargement ([problème #2](#)).

2.0.0 (09-09-2025)

Ajouté

- ▶ Macros `\intexgralsetup`, `\defaultdiff`, `\defaultvdiff` et `\vdiffstyle`.

Modifié

- ▶ Messages d'avertissement liés aux symboles non-existants. Ils ne sont maintenant provoqués que lorsque l'intégrale est composée.

Supprimé

- ▶ Clé `diff-vec-style` en faveur de `\vdiffstyle`.
- ▶ Message d'avertissement lié à la mauvaise utilisation des points-virgules en conjonction de la clé `int-split`.

Corrigé

- ▶ Bug où l'intégrande n'était pas réinitialisée lorsque `\integral` était employée successivement dans le même groupe \TeX ([détails ici](#)).

- Des restes de log *expl3* qui n'auraient pas dû être là.

1.1.0 (29-07-2025)

Ajouté

- Variantes étoilées pour les clés controlant symbole et borne (clés *single*, *contour* etc).

1.0.0 (26-07-2025) — Version initiale.

Index

Symbols

\! 742, 782, 830
 \: 743
 \{ 419, 434, 459, 461
 \} 419, 434, 459, 461
 _ . 97, 101, 105, 109, 113, 117, 122, 150, 152

B

bool commands:

\bool_if:NTF 294, 352, 369, 409, 485, 493, 549, 556, 572, 692, 713, 720, 728, 743, 746, 755, 764, 769, 774, 776, 786, 790, 793, 797, 808, 816, 833, 1020, 1072, 1086, 1101, 1115
 \bool_if:nTF 645, 735, 748, 821, 835
 \bool_new:N 39, 40, 41, 201, 202, 203, 204, 205, 206, 207, 208
 \bool_set_false:N 210, 211, 212, 213, 214, 215, 718, 869
 \bool_set_true:N
 .. 421, 437, 442, 717, 874, 879, 990, 992, 1011, 1013, 1015

C

\c 716
 \cdot 743, 755

clist commands:

\clist_item:Nn .. 338, 340, 356, 358, 364, 366, 373, 379, 597
 \clist_map_inline:Nn 551
 \clist_new:N 217, 218
 \clist_put_right:Nn 553
 \clist_reverse:N .. 1075, 1089, 1104, 1118
 \clist_set:Nn 335, 350, 398, 422, 445, 471, 596, 997, 1006, 1074, 1088, 1103, 1117
 \l_tmpa_clist .. 335, 338, 340, 350, 356, 358, 364, 366, 373, 379, 398, 401, 422, 426, 445, 449, 471, 472, 596, 597, 1074, 1075, 1076, 1088, 1089, 1090, 1103, 1104, 1105, 1117, 1118, 1119

cs commands:

\cs_generate_variant:Nn . 172, 173, 174, 175, 176, 177, 178
 \cs_if_exist:NTF 35, 904

\cs_new:Nn 86
 \cs_new:Npn 312, 407, 590
 \cs_new_protected:Nn . 267, 331, 660, 688, 726, 762, 806, 847
 \cs_new_protected:Npn 245, 248, 255, 292, 307, 321, 346, 392, 416, 431, 454, 469, 547, 594, 611
 \cs_set:Npn 1019
 \cs_set_eq:NN 779
 \cs_set_protected:Npn 730, 766, 810

D

\d 419, 434, 459, 461
 \DeclareLimitsKeyword 1111
 \DeclareSymbolKeyword 1219
 \DeclareVariableKeyword 1153
 \differentials 150, 152, 730, 766, 779, 810

E

exp commands:

\exp_args:Ne 950, 971
 \exp_args:NV ... 275, 563, 579
 \exp_args:NVV 558, 574
 \exp_last_unbraced:NV ... 602
 \exp_not:N 955, 976
 \exp_not:n 538
 \ExplSyntaxOff 1296

F

\frac 1271

G

group commands:

\group_begin: 1237
 \group_end: 1245

H

hook commands:

\hook_gput_code:nnn 34

I

\iiiint 1258
 \iiint 1257
 \iint 1256
 \infty ... 375, 381, 1265, 1266, 1267
 \int 131, 199, 908, 924, 1255

int commands:

\int_compare:nNnTF .. 495, 617, 623, 662, 667, 916, 925

<code>\int_compare_p:nNn</code>	647		
<code>\int_gincr:N</code>	690		
<code>\int_gzero_new:N</code>	241		
<code>\int_new:N</code>	240		
<code>\int_set:Nn</code>	269		
<code>\int_step_inline:nn</code> .	276, 771,		
	818, 921		
<code>\int_use:N</code>	89		
<code>\integral</code>	1235		
<code>\IntegralSetup</code>	1232, 1286		
integral internal commands:			
<code>__integragl_check_sequence_-</code>			
<code>size:</code>	660,		
	721		
<code>\l__integragl_custom_variables_-</code>			
<code>bool</code>	202, 212, 556, 572,		
	990		
<code>\l__integragl_deactivate_-</code>			
<code>differentials_bool</code>	40, 713,		
	992, 1060		
<code>\l__integragl_default_differential_-</code>			
<code>tl</code>	194, 581, 621, 1039,		
	1042		
<code>\l__integragl_default_vector_-</code>			
<code>differential_tl</code> . .	195, 567,		
	1052, 1055		
<code>\g__integragl_differential_-</code>			
<code>group_keyword_prop</code> . .	238,		
	394, 995, 1034, 1047, 1129, 1137,		
	1148, 1155		
<code>\l__integragl_differential_-</code>			
<code>options_i_tl</code>	188, 559, 564,		
	575, 580, 1026		
<code>\l__integragl_differential_-</code>			
<code>options_ii_tl</code>	189, 483, 487,		
	501, 515, 527, 538, 541		
<code>\l__integragl_differential_-</code>			
<code>order_i_seq</code>	229, 420,		
	423		
<code>\l__integragl_differential_-</code>			
<code>order_ii_seq</code> . . .	230, 425,		
	490		
<code>\l__integragl_differential_-</code>			
<code>seq</code> .	223, 535, 670, 675, 682,		
	731, 742, 756, 767, 781, 799, 829,		
	841, 857		
<code>\l__integragl_differential_-</code>			
<code>var_i_seq</code>	227, 436, 441, 446,		
	496, 506, 510		
<code>\l__integragl_differential_-</code>			
<code>var_ii_seq</code> . .	228, 448, 520,		
	524		
<code>__integragl_differentials:nn</code>			
.	407, 537, 1019		
<code>\l__integragl_display_mode_-</code>			
<code>str</code>	233, 850, 870, 875,		
	880		
<code>\l__integragl_domain_char_tl</code>			
. .	192, 949, 956, 970, 977		
<code>\l__integragl_domain_dimension_-</code>			
<code>tl</code>	193, 951, 957, 972,		
	978		
<code>\l__integragl_domain_seq</code>	231,		
	944, 945, 965, 966		
<code>\l__integragl_domain_style_tl</code>			
. . .	184, 955, 976, 1058		
<code>__integragl_extract_differential_-</code>			
<code>order:n</code>	416,		
	475		
<code>__integragl_extract_differential_-</code>			
<code>var:n</code>	431,		
	476		
<code>__integragl_extract_first_-</code>			
<code>key_name:n</code>	594,		
	1240		
<code>__integragl_generate_integral_-</code>			
<code>sequence:</code>	267, 888,		
	895		
<code>\l__integragl_has_order_bool</code>			
. . . .	207, 214, 421, 485		
<code>\l__integragl_has_var_bool</code> . .			
. .	208, 215, 437, 442, 493		
<code>\l__integragl_independent_-</code>			
<code>differential_options_tl</code> .			
. . . .	191, 456, 464, 484		
<code>\l__integragl_inline_symbol_-</code>			
<code>muskip</code>	235, 734,		
	1059		
<code>\g__integragl_integral_-</code>			
<code>number_int</code>	89, 241,		
	690		
<code>__integragl_integral_preconfiguration:</code>			
.	688, 849		
<code>\l__integragl_integral_-</code>			
<code>sequence_int</code> . . .	240, 269,		
	276		
<code>\l__integragl_integral_-</code>			
<code>symbol_seq</code>	221,		
	278, 663, 673, 680, 700, 702,		
	733, 771, 773, 818, 820		
<code>\l__integragl_integral_-</code>			
<code>symbol_tl</code>	187, 199, 257, 905,		

908, 919, 923, 927
 \backslash __intexgral_integrand_seq .
 . 220, 665, 668, 674, 681, 695,
 706, 709, 745, 785, 792, 832
 \backslash __intexgral_integrand_tl 183,
 697, 710, 716, 1243
 \backslash __intexgral_invert_differentials_-
 bool 41, 52, 737, 750, 774, 790,
 823, 837
 \backslash __intexgral_invert_limits_-
 bool .. 39, 48, 294, 352, 369,
 1072, 1086, 1101, 1115
 \backslash __intexgral_jacobian_bool .
 205, 746, 786, 793, 833, 1011
 \backslash __intexgral_jacobian_seq ..
 222, 747, 787, 794, 834, 1001
 \backslash __intexgral_key_name_tl 182,
 600, 606, 613
 \backslash c__intexgral_left_bracket_tl
 196, 377, 381
 \backslash g__intexgral_limits_keyword_-
 prop 237, 308,
 314, 315, 1069, 1084, 1099, 1113,
 1167, 1185, 1204
 \backslash __intexgral_limits_seq 224,
 326, 333, 348
 \backslash __intexgral_limits_style_tl
 42, 58, 60, 258
 \backslash __intexgral_lower_limit_tl
 185, 260, 296,
 300, 354, 362, 378, 898, 947,
 948, 953, 968, 969, 974, 984
 \backslash __intexgral_lower_limits_-
 seq .. 226, 271, 273, 281, 324,
 337, 371
 \backslash __intexgral_manual_differentials_-
 bool 201,
 211, 717, 718, 728, 739, 752, 764,
 776, 797, 808, 825, 839
 \backslash __intexgral_mathchoice:nnnn
 248, 929
 \backslash __intexgral_mkern:N . 245, 734
 \backslash __intexgral_new_differential_-
 group:nnn .. 392, 1132, 1141,
 1150, 1159
 \backslash __intexgral_new_limits_-
 group:nn 173, 307,
 1076, 1078, 1090, 1092, 1105,
 1107, 1119, 1121
 \backslash __intexgral_parse_differentials:
 547, 714
 \backslash __intexgral_parse_integral_-
 limit:n 312, 336,
 351
 \backslash __intexgral_parse_integral_-
 symbol: 255, 283,
 703
 \backslash __intexgral_parse_limits:n .
 321, 886, 893
 \backslash __intexgral_parse_macro_-
 keys:n 611,
 1241
 \backslash __intexgral_parse_variable:nn
 .. 469, 558, 563, 574, 579
 \backslash __intexgral_print_default_-
 integral: 726, 852,
 856
 \backslash __intexgral_print_integral:
 847, 1244
 \backslash __intexgral_print_nested_-
 integral: 762,
 853
 \backslash __intexgral_print_product_-
 integral: 806,
 854
 \backslash __intexgral_remove_differential_-
 order_and_var:n 454,
 477
 \backslash __intexgral_retrieve_key_-
 name:w 590,
 603
 \backslash c__intexgral_right_bracket_-
 tl 197, 375,
 383
 \backslash __intexgral_separate_-
 integral_bool 203, 213, 692,
 720, 869, 874, 879
 \backslash __intexgral_set_limits:nn ..
 280, 292
 \backslash __intexgral_set_limits_-
 regular: 331,
 887
 \backslash __intexgral_set_limits_-
 starred: 346,
 894
 \backslash __intexgral_starred_differential_-
 bool 206, 409, 1015,
 1020
 \backslash __intexgral_upper_limit_tl
 .. 186, 262, 297, 301, 900
 \backslash __intexgral_upper_limits_-
 seq . 225, 272, 282, 325, 339,

370
`\l__intexgral_variable_i-`
`clist` ... 217, 551, 576, 997,
 1006
`\l__intexgral_variable_ii-`
`clist` 218, 553,
 560
`\l__intexgral_vectorial_-`
`differential_bool` 204, 210,
 549, 743, 755, 769, 816, 1013
`\l__intexgral_vectorial_-`
`differential_style_tl` 190,
 554, 566, 1057
`__intexgral_warning_msg_-`
`header`: 86, 122, 129, 136, 143,
 151, 159, 165
`\intexgralsetup` 1250
iow commands:
`\iow_newline`: 22, 23, 92

K

keys commands:

`\keys_define:nn` . 46, 862, 1029,
 1172, 1189, 1208, 1221
`\keys_if_exist:nnTF` . 176, 613,
 1169, 1187, 1206
`\keys_set:nn` ... 614, 628, 1233

M

`\mathbb` 32, 35, 1291

`\mathchoice` 249

msg commands:

`\msg_critical:nn` 30
`\msg_error:nnn` 1070, 1094, 1131,
 1143, 1170, 1199
`\msg_line_context`: 90
`\msg_new:nnn` 19, 119, 127, 134, 141,
 148, 157, 163
`\msg_new:nnnn` . 95, 99, 103, 107,
 111, 115
`\msg_see_documentation_text:n`
 24
`\msg_warning:nn` 778, 917
`\msg_warning:nnn` 770, 817, 907,
 1004, 1168, 1186, 1205
`\msg_warning:nnnnn` .. 672, 679

muskip commands:

`\muskip_new:N` 235

N

`\NeedsTeXFormat` 3

`\NewDocumentCommand` 1067, 1082, 1097,
 1111, 1127, 1135, 1146, 1153, 1165,
 1183, 1202, 1219, 1232, 1235

`\NewLimitsKeyword`
 ... 117, 122, 1067, 1264, 1265,
 1266, 1267, 1268, 1269, 1270,
 1271, 1272, 1273, 1274, 1275

`\NewSymbolKeyword`
 .. 101, 1165, 1255, 1256, 1257,
 1258, 1259, 1260, 1261

`\NewVariableKeyword` 1127, 1278, 1279,
 1280, 1281, 1282

`\NewVarKeyword` 109

O

`\odif` 405, 410, 411

`\oiint` 1261

`\oint` 1260

`\oint` 1259

P

`\partial` 984

`\PassOptionsToPackage` 67, 69, 73, 75

`\phi` 1282

`\pi` 1269, 1270, 1271

`\pkginfograbProvidesExplPackage` 9

`\ProcessKeyOptions` 80

prop commands:

`\prop_get:NnNTF` 994, 1033, 1046,
 1069, 1099, 1129, 1148

`\prop_if_in:NnTF` 314, 1167, 1185,
 1204

`\prop_item:Nn` 315

`\prop_new:N` 237, 238

`\prop_pop:NnNTF` 1084, 1113, 1137,
 1155

`\prop_put:Nnn` 308, 394

`\ProvideLimitsKeyword` 1097

`\ProvideSymbolKeyword` 1202

`\ProvideVariableKeyword` 1146

Q

quark commands:

`\q_stop` 590, 603

R

regex commands:

`\regex_extract_once:nnNTF` 418,
 433

`\regex_if_match:nnTF` . 172, 716

`\regex_replace_all:nnN` .. 457

`\regex_split:nnN` 616

`\RenewLimitsKeyword` 113, 1082

`\RenewSymbolKeyword` 97, 1183
`\RenewVariableKeyword` 1135
`\RenewVarKeyword` 105
`\RequirePackage` 4, 5, 35, 82

S

`\s` 419, 434, 459, 461
 scan commands:
 `\scan_stop:` . 246, 779, 930, 931,
 932, 933
 seq commands:
 `\seq_clear:N` 324, 325, 1140, 1158
 `\seq_count:N` 272, 273, 496, 617,
 623, 647, 663, 665, 668, 670,
 673, 674, 675, 680, 681, 682, 771,
 818
 `\seq_gclear:N` 857
 `\seq_gpop_left:NN` 812
 `\seq_if_empty:NTF` 271, 700, 706
 `\seq_if_exist:NTF` 998
 `\seq_if_in:NnTF` 519
 `\seq_item:Nn` 281, 282, 423, 446,
 490, 506, 509, 539, 627, 630,
 631, 634, 649, 773, 781, 785, 787,
 820, 828, 832, 834, 841
 `\seq_map_indexed_inline:Nn` . .
 348, 479
 `\seq_map_inline:Nn` . . 333, 945,
 966
 `\seq_new:N` 220, 221, 222, 223, 224,
 225, 226, 227, 228, 229, 230,
 231
 `\seq_pop_left:NN` 523
 `\seq_put_left:Nn` 441
 `\seq_put_right:Nn` 278, 337, 339,
 370, 371, 535, 619, 624, 702, 708
 `\seq_set_eq:NN` 1000
 `\seq_set_from_clist:NN` . . 399,
 424, 447, 472
 `\seq_set_split:Nnn` . 326, 625,
 694, 944, 965
 `\seq_use:Nn` 177,
 731, 733, 742, 745, 747, 756, 767,
 792, 794, 799
 `\l_tmpa_seq`
 . 472, 479, 539, 616, 617, 620,
 623, 624, 627, 630, 634
 `\l_tmpb_seq` . 625, 631, 647, 649
`\sin` 1282
 str commands:
 `\str_case:nnTF` 634, 850
 `\str_case_e:nnTF` 373, 379

`\str_if_eq:nnTF` . . 174, 175, 505,
 508, 991
`\str_if_eq_p:nn` 178, 649
`\str_if_in:nnTF` 439
`\str_new:N` 233
`\str_set:Nn` 870, 875, 880
`\str_uppercase:n` 950, 971

T

\TeX and $\LaTeX 2_{\epsilon}$ commands:
 `\@ifpackagelater` 29
 `\@onlypreamble` 1250
 tex commands:
 `\tex_left:D` 375, 377
 `\tex_limits:D` 58
 `\tex_mathop:D` . . 742, 782, 830
 `\tex_mathpunct:D` 357, 365
 `\tex_mkern:D` 246, 930, 931, 932,
 933
 `\tex_nolimits:D` 60
 `\tex_right:D` 381, 383
`\theta` 1280, 1281, 1282
`\times` 948, 969
 tl commands:
 `\c_space_tl` 89
 `\tl_clear:N` 541, 919
 `\tl_const:Nn` 196, 197
 `\tl_head:n` 950, 971
 `\tl_if_blank:nTF` 396
 `\tl_if_empty:NTF` 947, 968
 `\tl_if_empty:nTF` 473, 480, 1238
 `\tl_if_in:NnTF` 598
 `\tl_new:N` . 42, 182, 183, 184, 185,
 186, 187, 188, 189, 190, 191, 192,
 193, 194, 195
 `\tl_put_right:Nn`
 . 482, 487, 500, 514, 526, 923,
 927, 948, 953, 969, 974
 `\tl_set:Nn`
 . 296, 297, 300, 301, 323, 354,
 362, 456, 597, 600, 943, 949,
 951, 964, 970, 972, 984, 1042,
 1055, 1243
 `\tl_set_eq:NN` . 58, 60, 199, 606,
 905, 908, 1038, 1051
 `\tl_tail:n` 952, 973
 `\tl_trim_spaces:n` 96, 100, 104,
 108, 112, 116, 130
 `\tl_use:N` . . . 257, 258, 378, 813
 `\l_tmpa_tl` 323, 326, 525, 597, 598,
 603, 606, 812, 813, 943, 944,
 964, 965, 995, 997, 1036, 1040,

1049, 1053, 1069, 1084, 1099, 1113, 1130, 1138, 1149, 1156	957
token commands:	<code>\token_to_str:N</code> 97, 101, 105, 109, 113, 122, 131, 150, 152
<code>\c_math_subscript_token</code> . 259, 978	V
<code>\c_math_superscript_token</code> 261,	<code>\vec</code> 1290

Implémentation

L'implémentation sera expliquée plus en détails dans une version ultérieure...

```
1 <*package>
2 <@@=intexgral>
3 \NeedsTeXFormat{LaTeX2e}
4 \RequirePackage{expl3}[2025-05-14]
5 \RequirePackage{pkginfograb}
6
7 % Package declaration
8
9 \pkginfograbProvidesExplPackage{intexgral}
10 {
11   name      = {intexgral} ,
12   author    = {Valentin Dao},
13   date      = {2025-12-24} ,
14   creation  = {2025-07-26} ,
15   version   = {3.0.0} ,
16   description = {A LaTeX package for typesetting integrals.}
17 }
18
19 \msg_new:nnn { intexgral } { expl3-too-old }
20 {
21   Your~expl3~installation~is~too~old,~
22   "intexgral"~requires~expl3~dated~2025-05-14~or~later.\iow_newline:
23   Package~loading~has~been~aborted.\iow_newline:
24   \msg_see_documentation_text:n { intexgral }
25 }
26
27 % expl3 version verification
28
29 \@ifpackagelater{expl3}{2025-05-14}{}
30 { \msg_critical:nn { intexgral } { expl3-too-old } }
31
32 % Verifies if \mathbb is defined (default domain style)
33
34 \hook_gput_code:nnn { begindocument/before } { . }
35 { \cs_if_exist:NF \mathbb { \RequirePackage{amsfonts} } }
36
37 % Definition of data types for package options
38
39 \bool_new:N \l__intexgral_invert_limits_bool
40 \bool_new:N \l__intexgral_deactivate_differentials_bool
41 \bool_new:N \l__intexgral_invert_differentials_bool
42 \tl_new:N \l__intexgral_limits_style_tl
43
44 % Package options
45
46 \keys_define:nn { intexgral }
47 {
48   invert-limits .bool_set:N = \l__intexgral_invert_limits_bool,
49   invert-limits .usage:n    = { preamble },
50   invert-limits .initial:n  = { false },
51
52   invert-diff .bool_set:N = \l__intexgral_invert_differentials_bool,
53   invert-diff .usage:n    = { preamble },
54   invert-diff .initial:n  = { false },
```

```

55
56     limits-mode .choice:,
57     limits-mode / limits .code:n =
58         { \tl_set_eq:NN \l__intexgral_limits_style_tl \tex_limits:D },
59     limits-mode / nolimits .code:n =
60         { \tl_set_eq:NN \l__intexgral_limits_style_tl \tex_nolimits:D },
61
62     limits-mode .usage:n = { preamble },
63     limits-mode .initial:n = { nolimits },
64
65     italic .choice:,
66     italic / true .code:n =
67         { \PassOptionsToPackage{italic=true}{derivative} },
68     italic / false .code:n =
69         { \PassOptionsToPackage{italic=false}{derivative} },
70
71     upright .choice:,
72     upright / true .code:n =
73         { \PassOptionsToPackage{upright=true}{derivative} },
74     upright / false .code:n =
75         { \PassOptionsToPackage{upright=false}{derivative} },
76 }
77
78 % Loads the package options and "derivative" package
79
80 \ProcessKeyOptions[intexgral]
81
82 \RequirePackage{derivative}
83
84 % Warning/Error/Info messages
85
86 \cs_new:Nn \__intexgral_warning_msg_header: {
87     (
88         integral~no.~
89         \int_use:N\g__intexgral_integral_number_int\c_space_tl
90         \msg_line_context:
91     )
92     \iow_newline:
93 }
94
95 \msg_new:nnnn { intexgral } { symb-key-alr-def }
96 { Symbol~key~"\tl_trim_spaces:n{#1}"~is~already~defined. }
97 { Use~\token_to_str:N \RenewSymbolKeyword\ instead. }
98
99 \msg_new:nnnn { intexgral } { symb-key-not-def }
100 { Symbol~key~"\tl_trim_spaces:n{#1}"~is~not~defined. }
101 { Use~\token_to_str:N \NewSymbolKeyword\ instead. }
102
103 \msg_new:nnnn { intexgral } { diff-group-alr-def }
104 { Differential~group~"\tl_trim_spaces:n{#1}"~is~already~defined. }
105 { Use~\token_to_str:N \RenewVarKeyword\ instead. }
106
107 \msg_new:nnnn { intexgral } { diff-group-not-def }
108 { Differential~group~"\tl_trim_spaces:n{#1}"~is~not~defined. }
109 { Use~\token_to_str:N \NewVarKeyword\ instead. }
110
111 \msg_new:nnnn { intexgral } { lim-group-alr-def }
112 { Limits~group~"\tl_trim_spaces:n{#1}"~is~already~defined. }

```

```

113 { Use~\token_to_str:N \RenewLimitsKeyword\ instead. }
114
115 \msg_new:nnnn { intexgral } { lim-group-not-def }
116 { Limits~group~"\tl_trim_spaces:n{#1}"~is~not~defined. }
117 { Use~\NewLimitsKeyword\ instead. }
118
119 \msg_new:nnn { intexgral } { key-exists-for-lim }
120 {
121   Symbol~key~already~defined~with~
122   \token_to_str:N \NewLimitsKeyword\ \__intexgral_warning_msg_header:
123   Key~"#1"~already~exists~for~predefined~limits.~
124   This~can~disrupt~the~functioning~of~the~special~syntax.
125 }
126
127 \msg_new:nnn { intexgral } { unknown-symb }
128 {
129   Unknown~integral~symbol~\__intexgral_warning_msg_header:
130   The~symbol~\tl_trim_spaces:n{#1}~has~been~replaced~with~
131   \token_to_str:N\int.
132 }
133
134 \msg_new:nnn { intexgral } { unequal-dim }
135 {
136   Inconsistent~structure~\__intexgral_warning_msg_header:
137   The~number~of~integrals~(#1),~integrands~(#2),~and~
138   differentials~(#3)~does~not~match.
139 }
140
141 \msg_new:nnn { intexgral } { no-jacobian }
142 {
143   Jacobian~unavailable~\__intexgral_warning_msg_header:
144   A~Jacobian~was~requested~for~the~"#1"~keyword,
145   ~but~none~was~declared~for~the~latter.
146 }
147
148 \msg_new:nnn { intexgral } { diff-not-sup }
149 {
150   Macro~\token_to_str:N\differentials\
151   not~supported~\__intexgral_warning_msg_header:
152   You~can't~use~\token_to_str:N\differentials\
153   with~"nested"~mode~alongside~inverted~differentials.~
154   I~will~simply~ignore~them.
155 }
156
157 \msg_new:nnn { intexgral } { diff-vec-not-supp }
158 {
159   Key~"diff-vec"~not~supported~\__intexgral_warning_msg_header:
160   You~can't~use~"diff-vec"~with~"#1"~mode.~I~will~simply~ignore~this~key.
161 }
162
163 \msg_new:nnn { intexgral } { use-glyph-instead }
164 {
165   Consider~using~the~dedicated~glyph~\__intexgral_warning_msg_header:
166   The~key~"nint"~was~used~with~fewer~than~
167   5~regular~integral~signs.
168 }
169
170 % Argument specifiers variants

```

```

171
172 \cs_generate_variant:Nn \regex_if_match:nnTF { nVTF }
173 \cs_generate_variant:Nn \__intexgral_new_limits_group:nn { nV }
174 \cs_generate_variant:Nn \str_if_eq:nnTF { neTF }
175 \cs_generate_variant:Nn \str_if_eq:nnF { neF }
176 \cs_generate_variant:Nn \keys_if_exist:nnTF { nVTF }
177 \cs_generate_variant:Nn \seq_use:Nn { Ne }
178 \cs_generate_variant:Nn \str_if_eq_p:nn { en }
179
180 % DATA TYPES
181
182 \tl_new:N \l__intexgral_key_name_tl
183 \tl_new:N \l__intexgral_integrand_tl
184 \tl_new:N \l__intexgral_domain_style_tl
185 \tl_new:N \l__intexgral_lower_limit_tl
186 \tl_new:N \l__intexgral_upper_limit_tl
187 \tl_new:N \l__intexgral_integral_symbol_tl
188 \tl_new:N \l__intexgral_differential_options_i_tl
189 \tl_new:N \l__intexgral_differential_options_ii_tl
190 \tl_new:N \l__intexgral_vectorial_differential_style_tl
191 \tl_new:N \l__intexgral_independent_differential_options_tl
192 \tl_new:N \l__intexgral_domain_char_tl
193 \tl_new:N \l__intexgral_domain_dimension_tl
194 \tl_new:N \l__intexgral_default_differential_tl
195 \tl_new:N \l__intexgral_default_vector_differential_tl
196 \tl_const:Nn \c__intexgral_left_bracket_tl { [ }
197 \tl_const:Nn \c__intexgral_right_bracket_tl { ] }
198
199 \tl_set_eq:NN \l__intexgral_integral_symbol_tl \int
200
201 \bool_new:N \l__intexgral_manual_differentials_bool
202 \bool_new:N \l__intexgral_custom_variables_bool
203 \bool_new:N \l__intexgral_separate_integral_bool
204 \bool_new:N \l__intexgral_vectorial_differential_bool
205 \bool_new:N \l__intexgral_jacobian_bool
206 \bool_new:N \l__intexgral_starred_differential_bool
207 \bool_new:N \l__intexgral_has_order_bool
208 \bool_new:N \l__intexgral_has_var_bool
209
210 \bool_set_false:N \l__intexgral_vectorial_differential_bool
211 \bool_set_false:N \l__intexgral_manual_differentials_bool
212 \bool_set_false:N \l__intexgral_custom_variables_bool
213 \bool_set_false:N \l__intexgral_separate_integral_bool
214 \bool_set_false:N \l__intexgral_has_order_bool
215 \bool_set_false:N \l__intexgral_has_var_bool
216
217 \clist_new:N \l__intexgral_variable_i_clist
218 \clist_new:N \l__intexgral_variable_ii_clist
219
220 \seq_new:N \l__intexgral_integrand_seq
221 \seq_new:N \l__intexgral_integral_symbol_seq
222 \seq_new:N \l__intexgral_jacobian_seq
223 \seq_new:N \l__intexgral_differential_seq
224 \seq_new:N \l__intexgral_limits_seq
225 \seq_new:N \l__intexgral_upper_limits_seq
226 \seq_new:N \l__intexgral_lower_limits_seq
227 \seq_new:N \l__intexgral_differential_var_i_seq
228 \seq_new:N \l__intexgral_differential_var_ii_seq

```

```

229 \seq_new:N \l__intexgral_differential_order_i_seq
230 \seq_new:N \l__intexgral_differential_order_ii_seq
231 \seq_new:N \l__intexgral_domain_seq
232
233 \str_new:N \l__intexgral_display_mode_str
234
235 \muskip_new:N \l__intexgral_inline_symbol_muskip
236
237 \prop_new:N \g__intexgral_limits_keyword_prop
238 \prop_new:N \g__intexgral_differential_group_keyword_prop
239
240 \int_new:N \l__intexgral_integral_sequence_int
241 \int_gzero_new:N \g__intexgral_integral_number_int
242
243 % TeX macros adapted to expl3 syntax
244
245 \cs_new_protected:Npn \__intexgral_mkern:N #1
246 { \tex_mkern:D #1 \scan_stop: }
247
248 \cs_new_protected:Npn \__intexgral_mathchoice:nnnn #1#2#3#4
249 { \mathchoice{#1}{#2}{#3}{#4} }
250
251 % SYMBOL
252
253 % General form of the symbol (glyph, limits mode, lower limit, upper limit)
254
255 \cs_new_protected:Npn \__intexgral_parse_integral_symbol:
256 {
257   \tl_use:N \l__intexgral_integral_symbol_tl
258   \tl_use:N \l__intexgral_limits_style_tl
259   \c_math_subscript_token
260   { \l__intexgral_lower_limit_tl }
261   \c_math_superscript_token
262   { \l__intexgral_upper_limit_tl }
263 }
264
265 % Generates the symbols after <limits> is executed
266
267 \cs_new_protected:Nn \__intexgral_generate_integral_sequence:
268 {
269   \int_set:Nn \l__intexgral_integral_sequence_int
270   {
271     \seq_if_empty:NTF \l__intexgral_lower_limits_seq
272     { \seq_count:N \l__intexgral_upper_limits_seq }
273     { \seq_count:N \l__intexgral_lower_limits_seq }
274   }
275   \exp_args:NV
276   \int_step_inline:nn \l__intexgral_integral_sequence_int
277   {
278     \seq_put_right:Nn \l__intexgral_integral_symbol_seq
279     {
280       \__intexgral_set_limits:nn
281       { \seq_item:Nn \l__intexgral_lower_limits_seq { ##1 } }
282       { \seq_item:Nn \l__intexgral_upper_limits_seq { ##1 } }
283       \__intexgral_parse_integral_symbol:
284     }
285   }
286 }

```

```

287
288 % LIMITS
289
290 % Inverts the limit order convention
291
292 \cs_new_protected:Npn \__intexgral_set_limits:nn #1#2
293 {
294   \bool_if:NTF \l__intexgral_invert_limits_bool
295   {
296     \tl_set:Nn \l__intexgral_lower_limit_tl { #2 }
297     \tl_set:Nn \l__intexgral_upper_limit_tl { #1 }
298   }
299   {
300     \tl_set:Nn \l__intexgral_lower_limit_tl { #1 }
301     \tl_set:Nn \l__intexgral_upper_limit_tl { #2 }
302   }
303 }
304
305 % Creates a keyword containing predefined limits
306
307 \cs_new_protected:Npn \__intexgral_new_limits_group:nn #1#2
308 { \prop_put:Nnn \g__intexgral_limits_keyword_prop { #1 } { #2 } }
309
310 % Verifies if the argument corresponds to a valid keyword
311
312 \cs_new:Npn \__intexgral_parse_integral_limit:n #1
313 {
314   \prop_if_in:NnTF \g__intexgral_limits_keyword_prop { #1 }
315   { \prop_item:Nn \g__intexgral_limits_keyword_prop { #1 } }
316   { #1 }
317 }
318
319 % Splits the argument of the key to prepare for parsing
320
321 \cs_new_protected:Npn \__intexgral_parse_limits:n #1
322 {
323   \tl_set:Nn \l_tmpa_tl { #1 }
324   \seq_clear:N \l__intexgral_lower_limits_seq
325   \seq_clear:N \l__intexgral_upper_limits_seq
326   \seq_set_split:NnV \l__intexgral_limits_seq { ; } \l_tmpa_tl
327 }
328
329 % Limits under regular form
330
331 \cs_new_protected:Nn \__intexgral_set_limits_regular:
332 {
333   \seq_map_inline:Nn \l__intexgral_limits_seq
334   {
335     \clist_set:Ne \l_tmpa_clist
336     { \__intexgral_parse_integral_limit:n { ##1 } }
337     \seq_put_right:Ne \l__intexgral_lower_limits_seq
338     { \clist_item:Nn \l_tmpa_clist { 1 } }
339     \seq_put_right:Ne \l__intexgral_upper_limits_seq
340     { \clist_item:Nn \l_tmpa_clist { 2 } }
341   }
342 }
343
344 % Limits under interval form

```

```

345
346 \cs_new_protected:Npn \__intexgral_set_limits_starred:
347 {
348   \seq_map_indexed_inline:Nn \l__intexgral_limits_seq
349   {
350     \clist_set:Nc \l_tmpa_clist
351     { \__intexgral_parse_integral_limit:n { ##2 } }
352     \bool_if:NTF \l__intexgral_invert_limits_bool
353     {
354       \tl_set:Nc \l__intexgral_lower_limit_tl
355       {
356         \clist_item:Nn \l_tmpa_clist { 2 }
357         \tex_mathpunct:D,
358         \clist_item:Nn \l_tmpa_clist { 1 }
359       }
360     }
361     {
362       \tl_set:Nc \l__intexgral_lower_limit_tl
363       {
364         \clist_item:Nn \l_tmpa_clist { 1 }
365         \tex_mathpunct:D,
366         \clist_item:Nn \l_tmpa_clist { 2 }
367       }
368     }
369     \bool_if:NTF \l__intexgral_invert_limits_bool
370     { \seq_put_right:Nc \l__intexgral_upper_limits_seq }
371     { \seq_put_right:Nc \l__intexgral_lower_limits_seq }
372   }
373   \str_case_e:nnF { \clist_item:Nn \l_tmpa_clist { 1 } }
374   {
375     { -\infty } { \tex_left:D \c__intexgral_right_bracket_tl }
376   }
377   { \tex_left:D \c__intexgral_left_bracket_tl }
378   \tl_use:N \l__intexgral_lower_limit_tl
379   \str_case_e:nnF { \clist_item:Nn \l_tmpa_clist { 2 } }
380   {
381     { +\infty } { \tex_right:D \c__intexgral_left_bracket_tl }
382   }
383   { \tex_right:D \c__intexgral_right_bracket_tl }
384 }
385 }
386 }
387
388 % VARIABLES
389
390 % Creates a keyword containing predefined variables
391
392 \cs_new_protected:Npn \__intexgral_new_differential_group:nnn #1#2#3
393 {
394   \prop_put:Nnn \g__intexgral_differential_group_keyword_prop
395   { #1 } { #2 }
396   \tl_if_blank:nF { #3 }
397   {
398     \clist_set:Nn \l_tmpa_clist { #3 }
399     \seq_set_from_clist:cN
400     { g__intexgral_#1_jacobian_seq }
401     \l_tmpa_clist
402   }

```

```

403 }
404
405 % expl3 equivalent of \odif(*)
406
407 \cs_new:Npn \__intexgral_differentials:nn #1#2
408 {
409     \bool_if:NTF \l__intexgral_starred_differential_bool
410     { \odif*[#1]{#2} }
411     { \odif[#1]{#2} }
412 }
413
414 % Extracts the whole "order" <keyval>
415
416 \cs_new_protected:Npn \__intexgral_extract_differential_order:n #1
417 {
418     \regex_extract_once:nnNT
419     { order\s*=\s*\{?\s*(\d+(?:\s*,\s*\d+)*)\s*\}? }
420     { #1 } \l__intexgral_differential_order_i_seq
421     { \bool_set_true:N \l__intexgral_has_order_bool }
422     \clist_set:Ne \l_tmpa_clist
423     { \seq_item:Nn \l__intexgral_differential_order_i_seq { 2 } }
424     \seq_set_from_clist:NN
425     \l__intexgral_differential_order_ii_seq
426     \l_tmpa_clist
427 }
428
429 % Extracts the whole "var" <keyval>
430
431 \cs_new_protected:Npn \__intexgral_extract_differential_var:n #1
432 {
433     \regex_extract_once:nnNTF
434     { var\s*=\s*(none|all|\{\s*(\d+(?:\s*,\s*\d+)*\s*)\}) }
435     { #1 }
436     \l__intexgral_differential_var_i_seq
437     { \bool_set_true:N \l__intexgral_has_var_bool }
438     {
439         \str_if_in:nnT { #1 } { var }
440         {
441             \seq_put_left:Nn \l__intexgral_differential_var_i_seq { var }
442             \bool_set_true:N \l__intexgral_has_var_bool
443         }
444     }
445     \clist_set:Ne \l_tmpa_clist
446     { \seq_item:Nn \l__intexgral_differential_var_i_seq { 3 } }
447     \seq_set_from_clist:NN
448     \l__intexgral_differential_var_ii_seq
449     \l_tmpa_clist
450 }
451
452 % Extracts all keys other than "order" and "var"
453
454 \cs_new_protected:Npn \__intexgral_remove_differential_order_and_var:n #1
455 {
456     \tl_set:Nn \l__intexgral_independent_differential_options_tl { #1 }
457     \regex_replace_all:nnN
458     {
459         order\s*=\s*\{?\s*\d+(\s*,\s*\d+)*\s*\}?\s*,?
460         |

```

```

461     [^-]var(?:\s*=\s*(?:none|all|\{\s*\d+(?:\s*,\s*\d+)*\s*\}))?,?
462   }
463   { }
464   \l__intexgral_independent_differential_options_tl
465 }
466
467 % Reassembles the variables
468
469 \cs_new_protected:Npn \__intexgral_parse_variable:nn #1#2
470 {
471   \clist_set:Nn \l_tmpa_clist { #2 }
472   \seq_set_from_clist:NN \l_tmpa_seq \l_tmpa_clist
473   \tl_if_empty:nF { #1 }
474   {
475     \__intexgral_extract_differential_order:n { #1 }
476     \__intexgral_extract_differential_var:n { #1 }
477     \__intexgral_remove_differential_order_and_var:n { #1 }
478   }
479   \seq_map_indexed_inline:Nn \l_tmpa_seq {
480     \tl_if_empty:nF { #1 }
481     {
482       \tl_put_right:NV
483         \__intexgral_differential_options_ii_tl
484         \__intexgral_independent_differential_options_tl
485       \bool_if:NT \__intexgral_has_order_bool
486       {
487         \tl_put_right:Ne \__intexgral_differential_options_ii_tl
488         {
489           ,order=
490           \seq_item:Nn \__intexgral_differential_order_ii_seq { ##1 }
491         }
492       }
493       \bool_if:NT \__intexgral_has_var_bool
494       {
495         \int_compare:nNnTF
496         { \seq_count:N \__intexgral_differential_var_i_seq }
497         =
498         { 1 }
499         {
500           \tl_put_right:Nn
501             \__intexgral_differential_options_ii_tl
502             { var }
503         }
504         {
505           \str_if_eq:neF { none }
506           { \seq_item:Nn \__intexgral_differential_var_i_seq { 2 } }
507           {
508             \str_if_eq:neTF { all }
509             { \seq_item:Nn
510               \__intexgral_differential_var_i_seq
511               { 2 }
512             }
513             {
514               \tl_put_right:Nn
515                 \__intexgral_differential_options_ii_tl
516                 { var }
517             }
518           }
519         }

```

```

519         \seq_if_in:NnT
520         \l__intexgral_differential_var_ii_seq
521         { ##1 }
522         {
523             \seq_pop_left:NN
524             \l__intexgral_differential_var_ii_seq
525             \l_tmpa_tl
526             \tl_put_right:Nn
527             \l__intexgral_differential_options_ii_tl
528             { ,var }
529         }
530     }
531 }
532 }
533 }
534 }
535 \seq_put_right:Ne \l__intexgral_differential_seq
536 {
537     \__intexgral_differentials:nn
538     { \exp_not:V \l__intexgral_differential_options_ii_tl }
539     { \seq_item:Nn \l_tmpa_seq { ##1 } }
540 }
541 \tl_clear:N \l__intexgral_differential_options_ii_tl
542 }
543 }
544
545 % Executes all previous macros to push tokens into the variable sequence
546
547 \cs_new_protected:Npn \__intexgral_parse_differentials:
548 {
549     \bool_if:NTF \l__intexgral_vectorial_differential_bool
550     {
551         \clist_map_inline:Nn \l__intexgral_variable_i_clist
552         {
553             \clist_put_right:Nn \l__intexgral_variable_ii_clist
554             { \l__intexgral_vectorial_differential_style_tl{##1} }
555         }
556         \bool_if:NTF \l__intexgral_custom_variables_bool
557         {
558             \exp_args:NVV \__intexgral_parse_variable:nn
559             \l__intexgral_differential_options_i_tl
560             \l__intexgral_variable_ii_clist
561         }
562         {
563             \exp_args:NV \__intexgral_parse_variable:nn
564             \l__intexgral_differential_options_i_tl
565             {
566                 \l__intexgral_vectorial_differential_style_tl
567                 { \l__intexgral_default_vector_differential_tl }
568             }
569         }
570     }
571 }
572 \bool_if:NTF \l__intexgral_custom_variables_bool
573 {
574     \exp_args:NVV \__intexgral_parse_variable:nn
575     \l__intexgral_differential_options_i_tl
576     \l__intexgral_variable_i_clist

```

```

577     }
578     {
579         \exp_args:NV \__intexgral_parse_variable:nn
580         \l__intexgral_differential_options_i_tl
581         { \l__intexgral_default_differential_tl }
582     }
583 }
584 }
585
586 % SPECIAL SYNTAX
587
588 % Cuts off the value of a key
589
590 \cs_new:Npn \__intexgral_retrieve_key_name:w #1=#2\q_stop { #1 }
591
592 % Extracts first key name of optional argument
593
594 \cs_new_protected:Npn \__intexgral_extract_first_key_name:n #1
595 {
596     \clist_set:Nn \l_tmpa_clist { #1 }
597     \tl_set:Ne \l_tmpa_tl { \clist_item:Nn \l_tmpa_clist { 1 } }
598     \tl_if_in:NnTF \l_tmpa_tl { = }
599     {
600         \tl_set:Ne \l__intexgral_key_name_tl
601         {
602             \exp_last_unbraced:NV
603             \__intexgral_retrieve_key_name:w \l_tmpa_tl \q_stop
604         }
605     }
606     { \tl_set_eq:NN \l__intexgral_key_name_tl \l_tmpa_tl }
607 }
608
609 % Sets the keys as usual, or sets them up manually if special syntax is in use
610
611 \cs_new_protected:Npn \__intexgral_parse_macro_keys:n #1
612 {
613     \keys_if_exist:nVTF { integral } \l__intexgral_key_name_tl
614     { \keys_set:nn { integral } { #1 } }
615     {
616         \regex_split:nnN { : } { #1 } \l_tmpa_seq
617         \int_compare:nNnT { \seq_count:N \l_tmpa_seq } < { 2 }
618         {
619             \seq_put_right:NV
620             \l_tmpa_seq
621             \l__intexgral_default_differential_tl
622         }
623         \int_compare:nNnT { \seq_count:N \l_tmpa_seq } < { 3 }
624         { \seq_put_right:Nn \l_tmpa_seq { d } }
625         \seq_set_split:Nne \l_tmpa_seq
626         { + }
627         { \seq_item:Nn \l_tmpa_seq { 2 } }
628         \keys_set:ne { integral }
629         {
630             limits = { \seq_item:Nn \l_tmpa_seq { 1 } },
631             variables = { \seq_item:Nn \l_tmpa_seq { 1 } },
632             mode =
633             {
634                 \str_case:enF { \seq_item:Nn \l_tmpa_seq { 3 } }

```

```

635         {
636             { d      } { default }
637             { n      } { nested  }
638             { p      } { product  }
639             { default } { default }
640             { nested  } { nested  }
641             { product } { product }
642         }
643     { default }
644 },
645 \bool_if:nT
646 {
647     \int_compare_p:nNn { \seq_count:N \l_tmpb_seq } > { 1 }
648     &&
649     \str_if_eq_p:en { \seq_item:Nn \l_tmpb_seq { 2 } } { j }
650 }
651 { jacobian }
652 }
653 }
654 }
655
656 % PRELIMINARY CONFIGURATIONS
657
658 % Verifies that <limits> and <variables> (and <integrand>) have the same size
659
660 \cs_new_protected:Nn \__intexgral_check_sequence_size:
661 {
662     \int_compare:nNnTF
663     { \seq_count:N \l__intexgral_integral_symbol_seq }
664     =
665     { \seq_count:N \l__intexgral_integrand_seq }
666     {
667         \int_compare:nNnF
668         { \seq_count:N \l__intexgral_integrand_seq }
669         =
670         { \seq_count:N \l__intexgral_differential_seq }
671         {
672             \msg_warning:nneee { intexgral } { unequal-dim }
673             { \seq_count:N \l__intexgral_integral_symbol_seq }
674             { \seq_count:N \l__intexgral_integrand_seq }
675             { \seq_count:N \l__intexgral_differential_seq }
676         }
677     }
678     {
679         \msg_warning:nneee { intexgral } { unequal-dim }
680         { \seq_count:N \l__intexgral_integral_symbol_seq }
681         { \seq_count:N \l__intexgral_integrand_seq }
682         { \seq_count:N \l__intexgral_differential_seq }
683     }
684 }
685
686 % Performs preparatory actions before executing the main macro
687
688 \cs_new_protected:Nn \__intexgral_integral_preconfiguration:
689 {
690     \int_gincr:N \g__intexgral_integral_number_int
691
692     \bool_if:NT \l__intexgral_separate_integral_bool

```

```

693     {
694         \seq_set_split:NnV
695         \l__intexgral_integrand_seq
696         { ; }
697         \l__intexgral_integrand_tl
698     }
699
700 \seq_if_empty:NT \l__intexgral_integral_symbol_seq
701 {
702     \seq_put_right:Nn \l__intexgral_integral_symbol_seq
703     { \__intexgral_parse_integral_symbol: }
704 }
705
706 \seq_if_empty:NT \l__intexgral_integrand_seq
707 {
708     \seq_put_right:NV
709     \l__intexgral_integrand_seq
710     \l__intexgral_integrand_tl
711 }
712
713 \bool_if:NF \l__intexgral_deactivate_differentials_bool
714 { \__intexgral_parse_differentials: }
715
716 \regex_if_match:nVTF { \c{differentials} } \l__intexgral_integrand_tl
717 { \bool_set_true:N \l__intexgral_manual_differentials_bool }
718 { \bool_set_false:N \l__intexgral_manual_differentials_bool }
719
720 \bool_if:NT \l__intexgral_separate_integral_bool
721 { \__intexgral_check_sequence_size: }
722 }
723
724 % Default mode
725
726 \cs_new_protected:Nn \__intexgral_print_default_integral:
727 {
728     \bool_if:NT \l__intexgral_manual_differentials_bool
729     {
730         \cs_set_protected:Npn \differentials
731         { \seq_use:Nn \l__intexgral_differential_seq { } }
732     }
733     \seq_use:Nn \l__intexgral_integral_symbol_seq
734     { \__intexgral_mkern:N \l__intexgral_inline_symbol_muskip }
735     \bool_if:nT
736     {
737         \l__intexgral_invert_differentials_bool
738         &&
739         !\l__intexgral_manual_differentials_bool
740     }
741     {
742         \seq_use:Nn \l__intexgral_differential_seq { } \tex_mathop:D{} \!
743         \bool_if:NT \l__intexgral_vectorial_differential_bool { \cdot \: }
744     }
745     \seq_use:Nn \l__intexgral_integrand_seq { }
746     \bool_if:NT \l__intexgral_jacobian_bool
747     { \seq_use:Nn \l__intexgral_jacobian_seq { } }
748     \bool_if:nT
749     {
750         !\l__intexgral_invert_differentials_bool

```

```

751      &&
752      !\l__intexgral_manual_differentials_bool
753    }
754    {
755      \bool_if:NT \l__intexgral_vectorial_differential_bool { \cdot }
756      \seq_use:Nn \l__intexgral_differential_seq { }
757    }
758  }
759
760 % Nested mode
761
762 \cs_new_protected:Nn \l__intexgral_print_nested_integral:
763 {
764   \bool_if:NT \l__intexgral_manual_differentials_bool
765   {
766     \cs_set_protected:Npn \differentials
767     { \seq_use:Nn \l__intexgral_differential_seq { } }
768   }
769   \bool_if:NT \l__intexgral_vectorial_differential_bool
770   { \msg_warning:nnn { intexgral } { diff-vec-not-supp } { nested } }
771   \int_step_inline:nn { \seq_count:N \l__intexgral_integral_symbol_seq }
772   {
773     \seq_item:Nn \l__intexgral_integral_symbol_seq { ##1 }
774     \bool_if:NTF \l__intexgral_invert_differentials_bool
775     {
776       \bool_if:NT \l__intexgral_manual_differentials_bool
777       {
778         \msg_warning:nn { intexgral } { diff-not-sup }
779         \cs_set_eq:NN \differentials \scan_stop:
780       }
781       \seq_item:Nn \l__intexgral_differential_seq { ##1 }
782       \tex_mathop:D{} \!
783     }
784     {
785       \seq_item:Nn \l__intexgral_integrand_seq { ##1 }
786       \bool_if:NT \l__intexgral_jacobian_bool
787       { \seq_item:Nn \l__intexgral_jacobian_seq { ##1 } }
788     }
789   }
790   \bool_if:NTF \l__intexgral_invert_differentials_bool
791   {
792     \seq_use:Nn \l__intexgral_integrand_seq {}
793     \bool_if:NT \l__intexgral_jacobian_bool
794     { \seq_use:Nn \l__intexgral_jacobian_seq { } }
795   }
796   {
797     \bool_if:NF \l__intexgral_manual_differentials_bool
798     {
799       \seq_use:Nn \l__intexgral_differential_seq {}
800     }
801   }
802 }
803
804 % Product mode
805
806 \cs_new_protected:Nn \l__intexgral_print_product_integral:
807 {
808   \bool_if:NT \l__intexgral_manual_differentials_bool

```

```

809     {
810         \cs_set_protected:Npn \differentials
811         {
812             \seq_gpop_left:NN \l_tmpa_tl
813             \tl_use:N \l_tmpa_tl
814         }
815     }
816     \bool_if:NT \l__intexgral_vectorial_differential_bool
817     { \msg_warning:nnn { intexgral } { diff-vec-not-supp } { product } }
818     \int_step_inline:nn { \seq_count:N \l__intexgral_integral_symbol_seq }
819     {
820         \seq_item:Nn \l__intexgral_integral_symbol_seq { ##1 }
821         \bool_if:nT
822         {
823             \l__intexgral_invert_differentials_bool
824             &&
825             !\l__intexgral_manual_differentials_bool
826         }
827         {
828             \seq_item:Nn
829             \l__intexgral_differential_seq { ##1 }
830             \tex_mathop:D{} \!
831         }
832         \seq_item:Nn \l__intexgral_integrand_seq { ##1 }
833         \bool_if:NT \l__intexgral_jacobian_bool
834         { \seq_item:Nn \l__intexgral_jacobian_seq { ##1 } }
835         \bool_if:nT
836         {
837             !\l__intexgral_invert_differentials_bool
838             &&
839             !\l__intexgral_manual_differentials_bool
840         }
841         { \seq_item:Nn \l__intexgral_differential_seq { ##1 } }
842     }
843 }
844
845 % General macro for typesetting
846
847 \cs_new_protected:Nn \__intexgral_print_integral:
848 {
849     \__intexgral_integral_preconfiguration:
850     \str_case:VnF \l__intexgral_display_mode_str
851     {
852         { default } { \__intexgral_print_default_integral: }
853         { nested } { \__intexgral_print_nested_integral: }
854         { product } { \__intexgral_print_product_integral: }
855     }
856     { \__intexgral_print_default_integral: }
857     \seq_gclear:N \l__intexgral_differential_seq
858 }
859
860 % KEYS
861
862 \keys_define:nn { integral } {
863
864     % LIMITS
865
866     mode .choice:,

```

```

867 mode / default .code:n =
868 {
869     \bool_set_false:N \l__intexgral_separate_integral_bool
870     \str_set:Nn \l__intexgral_display_mode_str { default }
871 },
872 mode / nested .code:n =
873 {
874     \bool_set_true:N \l__intexgral_separate_integral_bool
875     \str_set:Nn \l__intexgral_display_mode_str { nested }
876 },
877 mode / product .code:n =
878 {
879     \bool_set_true:N \l__intexgral_separate_integral_bool
880     \str_set:Nn \l__intexgral_display_mode_str { product }
881 },
882 mode .default:n = { default },
883
884 limits .code:n =
885 {
886     \__intexgral_parse_limits:n { #1 }
887     \__intexgral_set_limits_regular:
888     \__intexgral_generate_integral_sequence:
889 },
890
891 limits* .code:n =
892 {
893     \__intexgral_parse_limits:n { #1 }
894     \__intexgral_set_limits_starred:
895     \__intexgral_generate_integral_sequence:
896 },
897
898 llimit .tl_set:N = \l__intexgral_lower_limit_tl,
899
900 ulimit .tl_set:N = \l__intexgral_upper_limit_tl,
901
902 symbol .code:n =
903 {
904     \cs_if_exist:NTF #1
905     { \tl_set_eq:NN \l__intexgral_integral_symbol_tl #1 }
906     {
907         \msg_warning:nnn { intexgral } { unknown-symb } { #1 }
908         \tl_set_eq:NN \l__intexgral_integral_symbol_tl \int
909     }
910 },
911
912 % SYMBOL SHORTCUT
913
914 nint .code:n =
915 {
916     \int_compare:nNnT { #1 } < { 5 }
917     { \msg_warning:nn { intexgral } { use-glyph-instead } }
918
919     \tl_clear:N \l__intexgral_integral_symbol_tl
920
921     \int_step_inline:nn { #1 }
922     {
923         \tl_put_right:Nn \l__intexgral_integral_symbol_tl
924         { \int }

```

```

925     \int_compare:nNnT { ##1 } < { #1 }
926     {
927         \tl_put_right:Nn \l__intexgral_integral_symbol_tl
928         {
929             \__intexgral_mathchoice:nnnn
930             { \tex_mkern:D -12mu \scan_stop: }
931             { \tex_mkern:D -8mu \scan_stop: }
932             { \tex_mkern:D -4mu \scan_stop: }
933             { \tex_mkern:D -2mu \scan_stop: }
934         }
935     }
936 },
937
938
939 % LIMITS SHORTCUTS
940
941 domain .code:n =
942 {
943     \tl_set:Nn \l_tmpa_tl { #1 }
944     \seq_set_split:NnV \l__intexgral_domain_seq { * } \l_tmpa_tl
945     \seq_map_inline:Nn \l__intexgral_domain_seq
946     {
947         \tl_if_empty:NF \l__intexgral_lower_limit_tl
948         { \tl_put_right:Nn \l__intexgral_lower_limit_tl { \times } }
949         \tl_set:Ne \l__intexgral_domain_char_tl
950         { \exp_args:Ne \str_uppercase:n { \tl_head:n { ##1 } } }
951         \tl_set:Ne \l__intexgral_domain_dimension_tl
952         { \tl_tail:n { ##1 } }
953         \tl_put_right:Ne \l__intexgral_lower_limit_tl
954         {
955             \exp_not:N \l__intexgral_domain_style_tl
956             { \l__intexgral_domain_char_tl }
957             \c_math_superscript_token { \l__intexgral_domain_dimension_tl }
958         }
959     }
960 },
961
962 domain* .code:n =
963 {
964     \tl_set:Nn \l_tmpa_tl { #1 }
965     \seq_set_split:NnV \l__intexgral_domain_seq { * } \l_tmpa_tl
966     \seq_map_inline:Nn \l__intexgral_domain_seq
967     {
968         \tl_if_empty:NF \l__intexgral_lower_limit_tl
969         { \tl_put_right:Nn \l__intexgral_lower_limit_tl { \times } }
970         \tl_set:Ne \l__intexgral_domain_char_tl
971         { \exp_args:Ne \str_uppercase:n { \tl_head:n { ##1 } } }
972         \tl_set:Ne \l__intexgral_domain_dimension_tl
973         { \tl_tail:n { ##1 } }
974         \tl_put_right:Ne \l__intexgral_lower_limit_tl
975         {
976             \exp_not:N \l__intexgral_domain_style_tl
977             { \l__intexgral_domain_char_tl }
978             \c_math_subscript_token { \l__intexgral_domain_dimension_tl }
979         }
980     }
981 },
982

```

```

983 boundary .code:n =
984   { \tl_set:Nn \l__intexgral_lower_limit_tl { \partial #1 } },
985
986 % VARIABLES
987
988 variables .code:n =
989   {
990     \bool_set_true:N \l__intexgral_custom_variables_bool
991     \str_if_eq:nnTF { #1 } { none }
992       { \bool_set_true:N \l__intexgral_deactivate_differentials_bool }
993       {
994         \prop_get:NnNTF
995           \g__intexgral_differential_group_keyword_prop { #1 } \l_tmpa_tl
996           {
997             \clist_set:NV \l__intexgral_variable_i_clist \l_tmpa_tl
998             \seq_if_exist:cTF { g__intexgral_#1_jacobian_seq }
999               {
1000                 \seq_set_eq:Nc
1001                   \l__intexgral_jacobian_seq
1002                   { g__intexgral_#1_jacobian_seq }
1003               }
1004               { \msg_warning:nnn { intexgral } { no-jacobian } { #1 } }
1005           }
1006           { \clist_set:Nn \l__intexgral_variable_i_clist { #1 } }
1007       }
1008   },
1009
1010 jacobian .code:n =
1011   { \bool_set_true:N \l__intexgral_jacobian_bool },
1012 diff-vec .code:n =
1013   { \bool_set_true:N \l__intexgral_vectorial_differential_bool },
1014 diff-star .code:n =
1015   { \bool_set_true:N \l__intexgral_starred_differential_bool },
1016
1017 diff-symb .code:n =
1018   {
1019     \cs_set:Npn \__intexgral_differentials:nn ##1##2 {
1020       \bool_if:NTF \l__intexgral_starred_differential_bool
1021         { #1*{##1}{##2} }
1022         { #1[##1]{##2} }
1023     }
1024   },
1025
1026 diff-options .tl_set:N = \l__intexgral_differential_options_i_tl,
1027 }
1028
1029 \keys_define:nn { IntegralSetup }
1030 {
1031   defaultvar .code:n =
1032     {
1033       \prop_get:NnNTF
1034         \g__intexgral_differential_group_keyword_prop
1035         { #1 }
1036         \l_tmpa_tl
1037         {
1038           \tl_set_eq:NN
1039             \l__intexgral_default_differential_tl
1040             \l_tmpa_tl

```

```

1041     }
1042     { \tl_set:Nn \l__integragr_default_differential_tl { #1 } }
1043 },
1044 defaultvar* .code:n =
1045 {
1046     \prop_get:NnNTF
1047     \g__integragr_differential_group_keyword_prop
1048     { #1 }
1049     \l_tmpa_tl
1050     {
1051         \tl_set_eq:NN
1052         \l__integragr_default_vector_differential_tl
1053         \l_tmpa_tl
1054     }
1055     { \tl_set:Nn \l__integragr_default_vector_differential_tl { #1 } }
1056 },
1057 vectorstyle .tl_set:N = \l__integragr_vectorial_differential_style_tl,
1058 domainstyle .tl_set:N = \l__integragr_domain_style_tl,
1059 symbolskip .tl_set:N = \l__integragr_inline_symbol_muskip,
1060 hide-diff .bool_set:N = \l__integragr_deactivate_differentials_bool,
1061 }
1062
1063 % END USER MACROS
1064
1065 % Limits keywords
1066
1067 \NewDocumentCommand\NewLimitsKeyword{ m m }
1068 {
1069     \prop_get:NnNTF \g__integragr_limits_keyword_prop { #1 } \l_tmpa_tl
1070     { \msg_error:nnn { integragr } { lim-group-alr-def } { #1 } }
1071     {
1072         \bool_if:NTF \l__integragr_invert_limits_bool
1073         {
1074             \clist_set:Nn \l_tmpa_clist { #2 }
1075             \clist_reverse:N \l_tmpa_clist
1076             \__integragr_new_limits_group:nV { #1 } \l_tmpa_clist
1077         }
1078         { \__integragr_new_limits_group:nn { #1 } { #2 } }
1079     }
1080 }
1081
1082 \NewDocumentCommand\RenewLimitsKeyword{ m m }
1083 {
1084     \prop_pop:NnNTF \g__integragr_limits_keyword_prop { #1 } \l_tmpa_tl
1085     {
1086         \bool_if:NTF \l__integragr_invert_limits_bool
1087         {
1088             \clist_set:Nn \l_tmpa_clist { #2 }
1089             \clist_reverse:N \l_tmpa_clist
1090             \__integragr_new_limits_group:nV { #1 } \l_tmpa_clist
1091         }
1092         { \__integragr_new_limits_group:nn { #1 } { #2 } }
1093     }
1094     { \msg_error:nnn { integragr } { lim-group-not-def } }
1095 }
1096
1097 \NewDocumentCommand\ProvideLimitsKeyword{ m m }
1098 {

```

```

1099 \prop_get:NnNF \g__integragl_limits_keyword_prop { #1 } \l_tmpa_tl
1100 {
1101   \bool_if:NTF \l__integragl_invert_limits_bool
1102   {
1103     \clist_set:Nn \l_tmpa_clist { #2 }
1104     \clist_reverse:N \l_tmpa_clist
1105     \__integragl_new_limits_group:nV { #1 } \l_tmpa_clist
1106   }
1107   { \__integragl_new_limits_group:nn { #1 } { #2 } }
1108 }
1109 }
1110
1111 \NewDocumentCommand\DeclareLimitsKeyword{ m m }
1112 {
1113   \prop_pop:NnNT \g__integragl_limits_keyword_prop { #1 } \l_tmpa_tl
1114   {
1115     \bool_if:NTF \l__integragl_invert_limits_bool
1116     {
1117       \clist_set:Nn \l_tmpa_clist { #2 }
1118       \clist_reverse:N \l_tmpa_clist
1119       \__integragl_new_limits_group:nV { #1 } \l_tmpa_clist
1120     }
1121     { \__integragl_new_limits_group:nn { #1 } { #2 } }
1122   }
1123 }
1124
1125 % Variable keywords
1126
1127 \NewDocumentCommand\NewVariableKeyword{ m m o }
1128 {
1129   \prop_get:NnNTF \g__integragl_differential_group_keyword_prop { #1 }
1130   \l_tmpa_tl
1131   { \msg_error:nnn { integragl } { diff-group-alr-def } { #1 } }
1132   { \__integragl_new_differential_group:nnn { #1 } { #2 } { #3 } }
1133 }
1134
1135 \NewDocumentCommand\RenewVariableKeyword{ m m o }
1136 {
1137   \prop_pop:NnNTF \g__integragl_differential_group_keyword_prop { #1 }
1138   \l_tmpa_tl
1139   {
1140     \seq_clear:c { g__integragl_#1_jacobian_seq }
1141     \__integragl_new_differential_group:nnn { #1 } { #2 } { #3 }
1142   }
1143   { \msg_error:nnn { integragl } { diff-group-not-def } { #1 } }
1144 }
1145
1146 \NewDocumentCommand\ProvideVariableKeyword{ m m o }
1147 {
1148   \prop_get:NnNF \g__integragl_differential_group_keyword_prop { #1 }
1149   \l_tmpa_tl
1150   { \__integragl_new_differential_group:nnn { #1 } { #2 } { #3 } }
1151 }
1152
1153 \NewDocumentCommand\DeclareVariableKeyword{ m m o }
1154 {
1155   \prop_pop:NnNT \g__integragl_differential_group_keyword_prop { #1 }
1156   \l_tmpa_tl

```

```

1157     {
1158         \seq_clear:c { g__intexgral_#1_jacobian_seq }
1159         \__intexgral_new_differential_group:nnn { #1 } { #2 } { #3 }
1160     }
1161 }
1162
1163 % Symbol keys
1164
1165 \NewDocumentCommand\NewSymbolKeyword{ m m }
1166 {
1167     \prop_if_in:NnT \g__intexgral_limits_keyword_prop { #1 }
1168     { \msg_warning:nnn { intexgral } { key-exists-for-lim } { #1 } }
1169     \keys_if_exist:nnTF { integral } { #1 }
1170     { \msg_error:nnn { intexgral } { symb-key-alr-def } { #1 } }
1171     {
1172         \keys_define:nn { integral }
1173         {
1174             #1 .meta:n =
1175             {
1176                 symbol=#2,
1177                 llimit=##1
1178             },
1179         }
1180     }
1181 }
1182
1183 \NewDocumentCommand\RenewSymbolKeyword{ m m }
1184 {
1185     \prop_if_in:NnT \g__intexgral_limits_keyword_prop { #1 }
1186     { \msg_warning:nnn { intexgral } { key-exists-for-lim } { #1 } }
1187     \keys_if_exist:nnTF { integral } { #1 }
1188     {
1189         \keys_define:nn { integral }
1190         {
1191             #1 .undefine:
1192             #1 .meta:n =
1193             {
1194                 symbol=#2,
1195                 llimit=##1
1196             }
1197         }
1198     }
1199     { \msg_error:nnn { intexgral } { symb-key-not-def } }
1200 }
1201
1202 \NewDocumentCommand\ProvideSymbolKeyword{ m m }
1203 {
1204     \prop_if_in:NnT \g__intexgral_limits_keyword_prop { #1 }
1205     { \msg_warning:nnn { intexgral } { key-exists-for-lim } { #1 } }
1206     \keys_if_exist:nnF { integral } { #1 }
1207     {
1208         \keys_define:nn { integral }
1209         {
1210             #1 .meta:n =
1211             {
1212                 symbol=#2,
1213                 llimit=##1
1214             },

```

```

1215     }
1216   }
1217 }
1218
1219 \NewDocumentCommand\DeclareSymbolKeyword{ m m }
1220 {
1221   \keys_define:nn { integral }
1222   {
1223     #1 .undefine:
1224     #1 .meta:n =
1225     {
1226       symbol=#2,
1227       llimit=##1
1228     },
1229   }
1230 }
1231
1232 \NewDocumentCommand\IntegralSetup{ m }
1233 { \keys_set:nn { IntegralSetup } { #1 } }
1234
1235 \NewDocumentCommand\integral{ 0{} m }
1236 {
1237   \group_begin:
1238   \tl_if_empty:nF { #1 }
1239   {
1240     \__intexgral_extract_first_key_name:n { #1 }
1241     \__intexgral_parse_macro_keys:n { #1 }
1242   }
1243   \tl_set:Nn \l__intexgral_integrand_tl { #2 }
1244   \__intexgral_print_integral:
1245   \group_end:
1246 }
1247
1248 % Preamble only
1249
1250 \@onlypreamble\intexgralsetup
1251
1252 % Included keywords
1253
1254 % Symbol keywords
1255 \NewSymbolKeyword{single}      {\int}
1256 \NewSymbolKeyword{double}     {\iint}
1257 \NewSymbolKeyword{triple}     {\iiint}
1258 \NewSymbolKeyword{quadruple}  {\iiiiint}
1259 \NewSymbolKeyword{contour}    {\oint}
1260 \NewSymbolKeyword{surface}    {\oiint}
1261 \NewSymbolKeyword{volume}     {\oiint}
1262
1263 % Limits keywords
1264 \NewLimitsKeyword{ab}         {a, b}
1265 \NewLimitsKeyword{real}       {-\infty, \infty}
1266 \NewLimitsKeyword{positive}   {0, \infty}
1267 \NewLimitsKeyword{negative}   {-\infty, 0}
1268 \NewLimitsKeyword{unit}       {0, 1}
1269 \NewLimitsKeyword{circle}     {0, 2\pi}
1270 \NewLimitsKeyword{scircle}    {0, \pi}
1271 \NewLimitsKeyword{qcircle}    {0, \frac{\pi}{2}}
1272 \NewLimitsKeyword{height}     {0, H}

```

```

1273 \NewLimitsKeyword{radius}      {0, R}
1274 \NewLimitsKeyword{length}      {0, L}
1275 \NewLimitsKeyword{time}         {0, T}
1276
1277 % Variable keywords
1278 \NewVariableKeyword{cartesian}   {x, y, z}
1279 \NewVariableKeyword{planar}      {x, y}
1280 \NewVariableKeyword{polar}        {r, \theta}      [r]
1281 \NewVariableKeyword{cylindrical}  {r, \theta, z}    [r]
1282 \NewVariableKeyword{spherical}    {r, \theta, \phi} [r^2, \sin\theta]
1283
1284 % Default parameters
1285
1286 \IntegralSetup
1287 {
1288     defaultvar = {x},
1289     defaultvar* = {r},
1290     vectorstyle = \vec,
1291     domainstyle = \mathbb,
1292     symbolskip = {-6mu},
1293     hide-diff = false
1294 }
1295
1296 \ExplSyntaxOff

```