//这部分翻译自deal.II的github上的wiki页

**Rationale**

dealii中的绝大多数局部对象都有着张量积的结构。积分公式是一维积分公式的张量积。形函数是不同坐标方向上的多项式的张量积。甚至如Nedelec和Raviart-Thomas空间在各个分量上都有这种结构。Katharina Kormann和Martin Kronbichler已经证明了采用这些结构能显著地改善局部积分的效果。

为了优化局部积分，一条重要的要素是积分循环的长度在编译时就应已知。因此，积分点数量以及形函数数量必须作为模板参数。

**Tensor product quadrature**

**Tensor product structure of polynomials**

正如Katharina和Martin指出的那样，把张量积积分和张量积多项式结合起来，可以得到很高效的代码。

Example：Lagrange polynomials

给定次数为p-1的多项式以及每个方向上q个积分点，当前3D中的FEValues计算并存储p^3 q^3个形函数值，以及3倍数量的导数值和9倍数量的二阶导数值。

//以下部分翻译自Polynomials and polynomial spaces模块下的doxygen

**template<int dim, typename PolynomialType = Polynomials::Polynomial<double>>**

**class TensorProductPolynomials< dim, PolynomialType >**

这个类生成给定多项式的张量积

给定由n个一维多项式P1到Pn构成的向量，这个类生成ndim个形式为的多项式。如果基多项式在区间[-1,-1]或[0,1]上是互相正交的，则张量积多项式在[-1,1]dim或[0,1]dim上是正交的。

索引顺序如下：x坐标上的多项式先循环，然后y坐标的多项式，然后z的。 则前几项多项式为：P1(x)P1(y), P2(x)P1(y), P3(x)P1(y), ..., P1(x)P2(y), P2(x)P2(y), P3(x)P2(y), ...

output\_indices()函数可打印出dim维多项式的顺序，即，对于每个多项式，它会提供x，y，z方向上的一维多项式的指标i，j，k。dim维多项式的顺序可以通过函数set\_numbering()来改变。