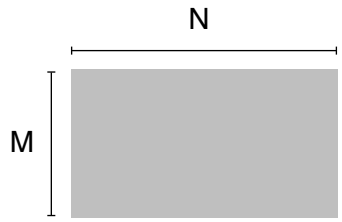
Document of FastAndroid

This is a document that introduce the api of our library — FastAndroid. In this library, we mainly implement two class, FaCollection and FaMatrix.

FaMatrix is a matrix operation library, including matrix transpose, matrix multiplication and matrix addition.

Before matrix representation in this library is wrapped in classes.



Class Name	Field	Description
MatrixInt	int[] data int M int N	This is class to represent a int type M*N matrix. The matrix data is stored in an int array.
MatrixFloat	float[] data int M int N	This is class to represent a float type M*N matrix. The matrix data is stored in an int array.
MatrixShort	short[] data int M int N	This is class to represent a short type M*N matrix. The matrix data is stored in an int array.

APIs	Input	Output	Description
public static void matrix_mul_int(MatrixInt a, MatrixInt b, MatrixInt result)	MatrixInt a and MatrixInt b used to multiplication. MatrixInt result for store the result	void	This api is used to matrix multiplication for integer data type.
public static void matrix_mul_float(MatrixFloat a, MatrixFloat b, MatrixFloat result)	MatrixFloat a and MatrixFloat b used to multiplication. MatrixFloat result for store the result	void	This api is used to matrix multiplication for float data type.
public static void matrix_mul_short(MatrixShort a, MatrixShort b, MatrixShort result)	MatrixShort a and MatrixShort b used to multiplication. MatrixShort result for store the result	void	This api is used to matrix multiplication for short data type.

APIs	Input	Output	Description
public static void matrix_transpose_int(MatrixInt a, MatrixInt result)	MatrixInt a used to transpose. MatrixInt result for store the result	void	This api is used to matrix transpose for integer data type.
public static void matrix_transpose_float(MatrixFloat a, MatrixFloat result)	MatrixFloat a used to transpose. MatrixFloat result for store the result	void	This api is used to matrix transpose for float data type.
public static void matrix_add_int(MatrixInt a, MatrixInt b, MatrixInt result)	MatrixInt a and MatrixInt b used to addition. MatrixInt result for store the result	void	This api is used to matrix addition for int data type.
public static void matrix_add_float(MatrixFloat a, MatrixFloat b, MatrixFloat result)	MatrixInt a and MatrixInt b used to addition. MatrixInt result for store the result	void	This api is used to matrix addtion for float data type.

FaCollection is a general tool box with three useful tools, vector operation, sorting and Fast Fourier transform. We design following apis:

APIs	Input	Output	Description
public static String test_vector()	void	A string of testing our vector operations.	This api call the vector operations itself and compare the results running on neon with the results running on c code. What's more, it will tell you whether the device is support neon operations.
public static String test_sort()	void	A string of testing our sorting operations.	This api call the sorting operations itself and compare the results running on neon, with the results running on c code (combsort and quicksort). What's more, it will tell you whether the device is support neon operations.
public static String test_fft()	void	A string of testing our fft operations.	This api call the fft operations itself and compare the results running on neon with the results running on c code. What's more, it will tell you whether the device is support neon operations.
public static void vector(Object[] array, Object[] vars)	the vectors in Object[], and parameters in Object[]	void, it change the input array in place	Detect the type of input array and run vector operations on that. In another word, calculate array[i] = sum_j (vars[j] * array[j]^j) where 0<j<vars.length. It support Float and Integer only.
public static void vector_int32(int[] array, int[] vars)	the vectors in int[], and parameters in int[]	void, it change the input array in place	Calculate array[i] = sum_j (vars[j] * array[j]^j) where 0<j<vars.length, for int type (both array and vars).

APIs	Input	Output	Description
public static void vector_float32(float[] array, float[] vars)	the vectors in float[], and parameters in float[]	void, it change the input array in place	Calculate array[i] = sum_j (vars[j] * array[j]^j) where 0<j<vars.length, for float type (both array and vars).
public static void qsort(Object[] array)	the unsorted array in Object[]	void, it change the input array in place	Detect the type of input array and sort it to ascending order using quick sort provided by standard C library. It support Float, Integer and Double only.
public static void qsort_int32(int[] array)	the unsorted array in int[]	void, it change the input array in place	Sort the array to ascending order using quick sort provided by standard C library for int type.
public static void qsort_float32(float[] array)	the unsorted array in float[]	void, it change the input array in place	Sort the array to ascending order using quick provided by standard C library for float type.
public static void qsort_double32(double[] array)	the unsorted array in double[]	void, it change the input array in place	Sort the array to ascending order using quick sort provided by standard C library for double type.
public static void sort(Object[] array)	public static void sort(Object[] array)	void, it change the input array in place	Detect the type of input array and sort it to ascending order using AA sort provided by standard C library. It support Float, Integer only. It direct the Double to quick sort for convenience.
public static void sort_int32(int[] array)	the unsorted array in int[]	void, it change the input array in place	Sort the array to ascending order using AA provided by standard C library for int type.
public static void sort_float32(float[] array)	the unsorted array in float[]	void, it change the input array in place	Sort the array to ascending order using AA provided by standard C library for float type.
public static void fft(Object[] real, Object[] imag)	the real part and Imaginary part in Object[]	void, it change the input array in place	Detect the type of input array and run FFT on the input values, but it need the length of real and imaginary equals to each other and it must be power of 2. It support Float only.
public static void fft_float32(float[] real, float[] imag)	the real part and Imaginary part in float[]	void, it change the input array in place	Run FFT on the input values, but it need the length of real and imaginary equals to each other and it must be power of 2. For float type.
public static void ifft(Object[] real, Object[] imag)	the real part and Imaginary part in Object[]	void, it change the input array in place	Detect the type of input array and run IFFT on the input values, but it need the length of real and imaginary equals to each other and it must be power of 2. It support Float only.
public static void ifft_float32(float[] real, float[] imag)	the real part and Imaginary part in float[]	void, it change the input array in place	Run IFFT on the input values, but it need the length of real and imaginary equals to each other and it must be power of 2. For float type.